Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems

November 11-15, 1991

KEK, Tsukuba, Japan

NATIONAL LABORATORY FOR HIGH ENERGY PHYSICS, KEK

FOREWORD

International Conference on Accelerators and Large Experimental Physics Control Systems (ICALEPCS'91) was held on November 11 - 15, 1991 at KEK, Tsukuba, Japan. This was the first conference in this series held in Asia. It was a great pleasure for the organizers of the conference that more than 240 participated. Among them we were delighted to see large delegates from countires and institutions such as Russia, China, Korea, India, SRRC in Hsinchu, etc. from which very few had participated in the former conferences. This reflects the fact that the society has come to its maturity, to which the continuing effort by EPCS(Interdivisional Group on Experimental Physics Control Systems under European Physical Society) has contributed greatly.

The maturity was also demonstrated by a naming of the "standard model" of control systems which we heard during the conference. Also, in the conference, discussion was held on prospects of making a generic tool-kit of control system on the basis of world-wide collaboration. Of course, we are far from this goal; however, it is sure that we glimpsed our future during the conference.

黒川真一
Shin-ichi Kurokawa
Chairperson of ICALEPCS'91

加藤直彦
Tadahiko Katoh
Chairperson of Local Organizing Committee

LIST OF INTERNATIONAL SCIENTIFIC ADVISORY COMMITTEE MEMBERS

BARTON, Donald S. *(BNL)*
BLUMER, Thomas *(PSI)*
BUSSE, Winffied *(HMI)*
CLOUT, Peter Norman *(VISTA)*
DANEELS, Axel Jean *(CERN)*
DASGUPTA, Subrata *(VECC)*
DOHAN, Don A. *(SSCL)*
DUNAITSEV, Anatoly F. *(IHEP-Serpukhov)*
GURD, David P. *(SSCL)*
HUMPHREY, John (Rusty)*(SLAC)*
JAN, Gwo-Jen *(SRRC)*
KATOH, Tadahiko *(KEK)*
KIMURA, Toyoaki *(JAERI-Naka)*
KUIPER, Berend *(CERN)*
KUPER, Edward *(INP-Novosibirsk)*
KUROKAWA, Shin-ichi *(KEK)* - Chairperson
LIU, Shi-Yao *(IHEP-Beijing)*
LUCAS, Peter Wayne *(Fermilab)*
LUDGATE, George Arthur *(TRIUMF)*
LUONG, Tam T. *(GANIL)*
MÜLLER, Klaus-Dieter *(KFA)*
NAVRATIL, Jiri *(CTU)*
NISHIMURA, Hiroshi *(LBL)*
OWEN, Edward Charles *(Daresbury)*
PETERS, Franz *(DESY)*
RAUSCH, Raymond *(CERN)*
SCHALLER, Stuart *(LANL)*
STEINER, Rudolf *(GSI)*
WADA, Takeshi *(RIKEN)*
WON, Sangchul *(PLS)*

LIST OF LOCAL ORGANIZING COMMITTEE MEMBERS

KATOH, Tadahiko *(KEK)* - Chairperson
KIMURA, Toyoaki *(JAERI-Naka)*
KOHNO, Toshiyuki *(NIRS)*
KUROKAWA, Shin-ichi *(KEK)*
MORITA, Koh-Ichiro *(NAO-NRO)*
NAKAHARA, Kazuo *(KEK)*
OGATA, Hiroshi *(RCNP)*
OYAMADA, Masayuki *(LNS)*
PAK, Cheol On *(KEK)*
TAKEDA, Shigeru *(KEK)*
URAKAWA, Junji *(KEK)*
WADA, Takeshi *(RIKEN)*
WATANABE, Shin-ichi *(INS)*

Organized by: KEK, National Laboratory for High Energy Physics

Supported by: European Physical Society
IEEE Nulcear and Plasma Sciences Society
Japan Physical Society

Sponsored by: Tsukuba EXPO '85 Memorial Foundation
Foundation for High Energy Accelerator Science

Contents

A USERS VIEW OF THE SPS AND LEP CONTROL SYSTEMS

R. Bailey
CERN
CH–1211 Geneva 23, Switzerland

Abstract

Every accelerator has a control system; at present the SPS has two, both of which are needed to run the machine. Consequently a user of the SPS / LEP complex has to be concurrently familiar with three control systems. While this situation brings problems it allows, even forces, comparison between the different systems, which in turn enriches the user viewpoint.

This paper assesses the SPS and LEP control systems from the point of view of the user, who may be an equipment specialist, operator, accelerator physicist or combinations thereof.

1. Introduction – what the accelerators do

Exploitation of the two large accelerators at CERN is a varied business. For the SPS in 1991 this amounts to running as a fixed target machine for over half the year, providing either protons (during 21 weeks) or sulphur ions (during 6 weeks) to the physics community. In conjunction with this the SPS acts as an injector to LEP, providing leptons in an interleaved repetitive supercycle. Furthermore about 15% of the fixed target running time is given over to machine development periods, when the SPS is required to run in some non–standard way, mostly as a testbed for the LHC. Finally, the SPS is also used in the other major mode of operation, as a proton–antiproton collider, for about 5 weeks.

In parallel with all of the 27 weeks of SPS fixed target running, LEP is taking beam either for Z^0 production or for a substantial machine development program, the latter amounting to about 30% of the total LEP running time.

For both machines, although mostly for LEP, installation and testing of new equipment is carried out throughout the year.

This diversity of operations and machine improvement is carried out from a common central control room, with the same teams being responsible for both the SPS and LEP. In particular, one group run the SPS in a variety of modes of operation throughout the year as well as running LEP. This means that these personnel have to be familiar with the different control systems used to interact with the accelerators. The same is true of the personnel responsible for equipment commissioning.

2. Overview of control systems available

From 1975 the SPS has been controlled, either exclusively or partially, via a system based on Norsk Data ND100 computers connected in a TITN star configuration [1]. The computers run SINTRON and the programmers are provided with the NODAL interpreter, libraries of graphics primitives and data modules and a means of calling FORTRAN executables [2].

From 1985 the major new requirement for SPS to provide beams to LEP meant a complete rewrite of the applications software. This was undertaken in a UNIX environment on an Apollo network, with C as the main programming language and Apollo–Dialog for the user interface. In the first instance access to the hardware was via a gateway into the existing TITN system. More recently the possibility exists to access some equipment completely independently from the TITN system, using the same overall Token Ring architecture as for LEP (see below).

Presently the SPS is run using a mixture of purely TITN (30%), Apollo via the gateway into TITN (50%), and purely Apollo Token Ring applications (20%) (see figure 1).

LEP applications also run on an Apollo network, with C as the main programming language and Apollo–Dialog for the user interface. All the Apollos are connected on a control room Token Ring, with communications out into the field through a bridge to a machine Token Ring running around the accelerator [3] . At several points around the ring there is a further bridge or gateway into either a regional Token Ring or an Ethernet network. Connected to these local area networks is a variety of configurations, allowing access into the hardware via several different equipment control assemblies, mostly using the MIL–1553–B standard (see figure 1).

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S01SRA01

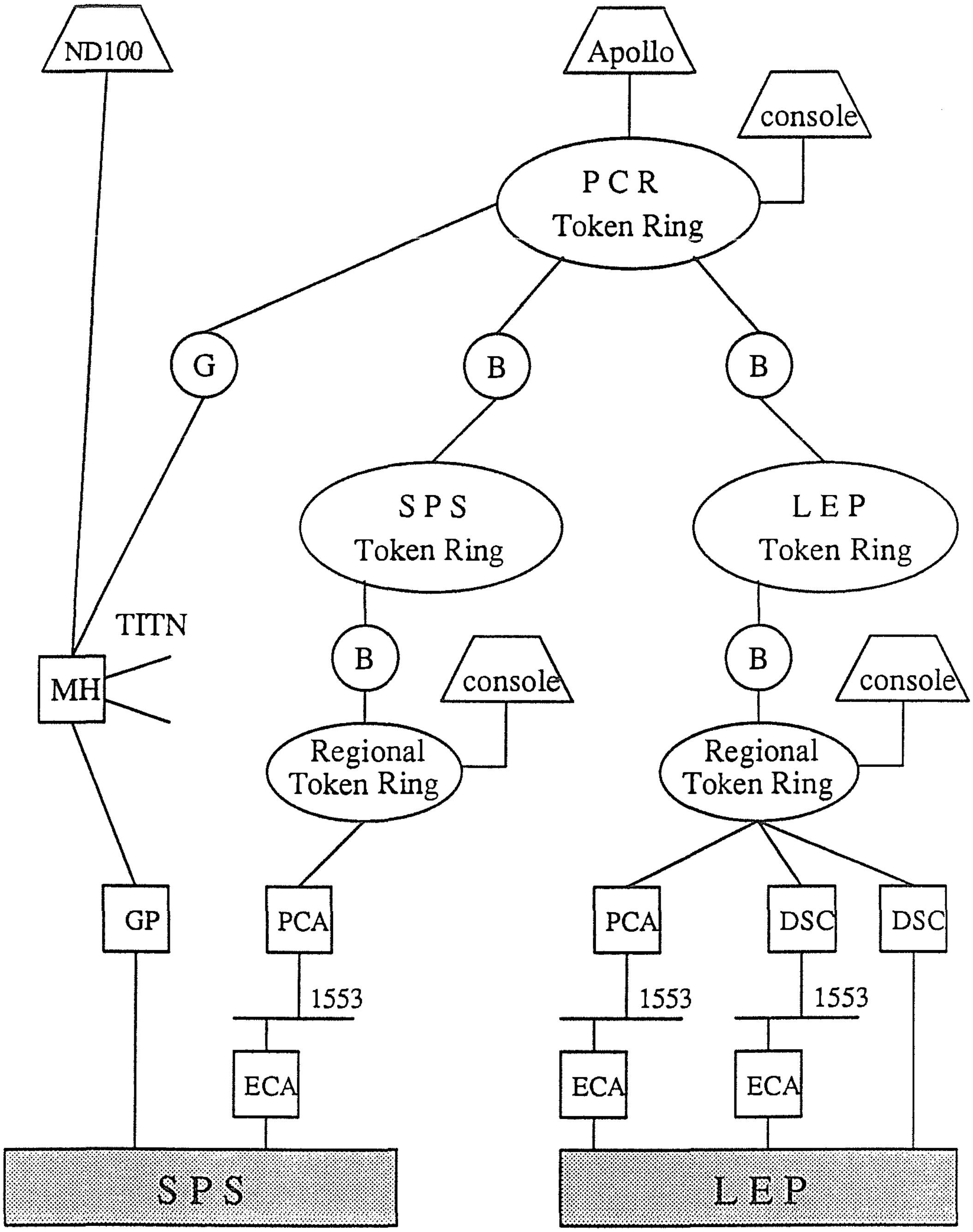


Figure 1
Logical schematic of the networks

3. Different types of user

The control systems of SPS and LEP are used at different times by a variety of different personnel. These largely divide into three categories; operators, accelerator physicists and equipment specialists, each of whom have somewhat different requirements for the control system. These requirements are not only for the underlying architecture (network, operating system etc), but also for the applications that run on top of it. In other words, the user here is seen as the person who runs the applications programs, rather than the person who writes them.

All types of user of course need reliable network communications, with good diagnostics when things go wrong. An adequate speed across the network from console to equipment is also generally required.

3.1 Equipment specialists

Equipment specialists need to access a diversity of accelerator hardware, setting and reading a multitude of parameters that are not of interest to other users of the control system. In many cases they also need to do this locally, in order to closely monitor the effects on their equipment. This means that they need to run specific programs both in the central control room and in the field, the latter requiring local console facilities. They may well want to run locally when the network is down. Most of these programs are written by the person who will run them, or at least by a close colleague, and as such the reliability of the application is not of great importance.

In many cases the amount of equipment accessed is far more than during normal operations, in order to thoroughly test a system, for example. For this reason the speed can be of prime importance to the equipment groups.

Key requirements;

local console facilities
execution speed

3.2 Operators

Operators rarely work on individual pieces of equipment, but rather on combinations of accelerator systems or even on the accelerator as a whole. In performing this work they prefer to see a high level of standardisation across the different applications and across the different accelerators. The applications also need to be easy to use, with the operator being presented with all the information that he needs but not swamped by auxiliary data that he rarely uses. Online documentation is a big help, particularly when the applications are new.

Since many tasks have to be performed at the right time in a sequence, the applications that perform them need to be highly reliable. Since operations is a long and repetitive process, it is essential that the speed of execution of programs is adequate, which generally means completion of the task in a matter of seconds. Good error reporting is also very important.

Key requirements;

ease of use
stability
standardisation
execution speed
error reporting

3.3 Accelerator physicists

Accelerator physicists have essentially the same requirements as the operators, except for the important addition of flexibility to allow new, non-standard applications to be used. Indeed since machine development periods usually involve doing several unusual things, standardisation and error reporting are not so important.

Key requirements;

flexibilty

ease of use
stability
execution speed

4. Comparison of the different control systems

Table 1 summarises the results discussed in more detail here.

In all three cases the speed and reliability of the network is adequate. However when there is a problem, it is much easier to pinpoint on the TITN system than on the Token Rings, which have become extremely complex.

Local facilities are also better on the TITN, where much of the equipment data is stored locally rather than in a central database.

Table 1 Comparison of observations

	SPS old	SPS new	LEP
Network			
Speed	●●	●●	●●
Reliability	●●	●●	●●
Diagnostics	●●●	●	●
Local facilities	●●●	●●	●●
Applications			
Execution speed	●●	●●	●
Stability	●●	●●●	●●
Error reporting	●	●●●	●
Standardisation	●	●●●	●●
Flexibility	●●●	●	●●
Ease of use	●●	●●●	●●●

Key The more blobs the better

● poor

●● adequate

●●● good

4.1 SPS old

A key feature of the NODAL based control system is flexibility. It is extremely easy to produce a working application program, communicating with the machine and displaying data to the user. While this is an excellent feature, particularly for equipment testing or for one–off applications, as operations become more complicated it becomes more difficult to control the overall coherence of the system.

In the SPS the operational applications grew out of equipment commissioning programs, essentially on a system by system basis, and in an iterative way. As an example quadrupoles, sextupoles, octupoles etc were all controlled by different suites of programs all essentially doing the same thing. Adding a new system involved adding a new suite of programs to control it. Apart from the obvious problems of duplication of effort, this has also led to a certain diversity of the way similar functions had to be performed in different applications, which is very confusing to the user and makes it difficult to remember how to drive the different programs.

Because it is so easy to write or modify programs in this environment, in the absence of any real software management the stability of the applications is never fully achieved, and maintenance is consequently very difficult.

The very limited memory available in the control room consoles meant that most of the applications had to be kept small, and as a direct consequence of this error reporting had to be kept to a minimum, as did commenting the code.

Finally the speed at which the applications run has been found to be adequate. Since no online database exists the individual programs do their own data management, and though this brings its own problems it tends to be fast. Consequently the speed is determined by that of the NODAL interpretor and that of the TITN network. As a benchmark, sending a 100 point amplitude vs time function to the accelerator takes around 1 second per hardware address, which is considered acceptable.

4.2 SPS new

There were two significant differences between the way the new SPS applications were developed as compared to the old. Firstly the overall functionality of the software needed to operate the accelerator was analysed in detail before any design was considered, and secondly the underlying data structures were completely determined before any implementation was undertaken [4]. By its very nature this kind of software development leads to software that needs little change once implemented, and results in a very stable system. The highly modular way in which the applications were designed allowed an easy and standard way of handling errors, and the error reporting is excellent.

Knowing the detailed functionality led to a high uniformity, not just at the level of the operator interface [5] but more generally in the facilities the different applications shared. As examples there is only one function editor, one dataviewer and one application that is able to send to the equipment anything from a single function to the settings for the whole machine. This has contributed greatly to the ease of use of the software, and this is enhanced by a standard online help facility describing how to

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S01SRA01

drive the applications.

Having a sound definition of the data has allowed the applications to be largely data driven, giving coherence to the different accelerator systems and allowing new systems to be integrated without writing a single word of code.

The major disadvantage of this approach is that the software has been produced specifically to operate the SPS in the various modes forseen over the next ten years. Any novel running of the accelerator during machine development sessions invariably requires new features which are very difficult, sometimes impossible, to accomodate. Up to now these problems have been overcome by exploiting the high flexibility of the old TITN system.

The speed of execution of tasks is similar to the old system, but in this case database access times and the TITN network are the determining factors. The reliability of the gateway into the TITN is not good but problems are easy to spot and rectify.

4.3 LEP

Before the construction of LEP was complete, an analysis of the software required to run the machine was made. Naturally the emphasis was put on the software needed to commission the accelerator, and for the startup of LEP the controls and equipment groups provided a suite of powerful utilities for sending settings to and acquiring data from the hardware. These utilities exist as commands on the control room consoles and provide a means of quickly making script programs to do standard or non-standard things to the accelerator. Much use of this facility has been made during the commissioning phase, and more recently by accelerator physicists during machine development sessions.

The applications used today in operations also make heavy use these utilities for accessing the hardware. While this may be convenient for the programmers it invariably introduces overheads in the execution speed. The speed is further reduced by the underlying online data organisation, since the structuring of the data does not reflect the way in which we now want to run the machine.

The development of the operational applications has not followed an integrated approach, which has brought low coherence and a very variable level of error reporting.

Uniformity across the user interface applications has been achieved to some extent. Following the standards of the SPS has ensured a look and feel of the individual applications that is liked by the operators, and most programs are now easy to use.

The operational software relies heavily on servers running at all levels of the network, from the control room Apollos down to the front end computers. While communication between these servers is normally transparent to the user it often involves passing through several bridges. If one of the bridges or servers dies it is sometimes difficult to diagnose which one, and in many cases a procedure of sequentially restarting one after the other is required.

Furthermore many applications are dependant on certain computers to be up in order to run. There are presently around 10 such critical nodes on the control room token ring, the failure of any one of which would affect operations to some extent, in many cases seriously.

These two implementation details directly affect the overall stability of the software needed to run the machine.

5. Alarms

Quite apart from the application software used to drive the accelerator, there is another area of the control system that is of great importance during routine operation of the machine, namely the surveillance system.

Ideally this should work on the simple principle that software, running without operator intervention, should check that all elements required to be ON are ON, that those that should be OFF are OFF, and that all settings stay within a tolerance acceptable for operations. This software, running frequently, should report any abnormal findings to an alarm system for processing prior to presentation to the operator as a new alarm on his screen [6].

In practice the viability of such a system depends very much on other parts of the control system. It is imperative for such a system to have available a definitive source of data reflecting the way the machine is actually supposed to run at the time. Furthermore because most machines run in several modes of operation, each requiring a different configuration, this image of the machine has to be dynamic.

It has already been mentioned that the LEP operational applications have not been developed in an integrated way, and one consequence of this is that there are several different ways of storing the actual machine settings. This makes it very difficult to provide standard surveillance programs; in reality each set of equipment has to have it's own program, a situation which is of course very difficult to administrate. So while for LEP the central alarm server works well, the amount of useful information reaching the operator is presently rather limited.

The same problems were encountered with the alarm system running on the SPS TITN network. Again there was no coherent image of the machine, and it took several years before the alarm system was providing information of sufficient credibility for the operators to use with confidence.

It is ironic that the new SPS operational software, which is driven from a central online database, does not yet have any kind of alarm system. Indeed we are experiencing problems due to this as more and more systems are migrated from the TITN to the Apollo–based software, since there is presently no means of surveying them. The aim is eventually to use the same system that is presently in use for LEP, but with simple surveillance programs comparing measurements with settings in the online database.

6. Conclusions and remarks

In the case of both the SPS and LEP, the network and control room utilities proved adequate during the running in of the machine. As testimony to this, beam was circulating in LEP one or two days after first injection, and the first Z^0 was reported within a month.

However, remember also that machine commissioning is done by specialists and over a limited period of time. When it comes to building the complex, integrated software packages that are required in routine operations, it has proved difficult to do so from the utilities provided. What is needed is a review of the operational requirements and a corresponding rewrite of the application software. Furthermore it is very difficult to determine these operational requirements in advance of getting hands–on experience of the accelerator.

The new SPS software is a good example of what can be done. It was based on 10 years experience of running the SPS in a variety of modes, and the software produced satisfies most of the operational requirements.

The same thing now has to be done for LEP, this time after 3 years experience but drawing on the lessons learned in the SPS.

7. References

[1] M.C. Crowley–Milling, "The Design of the Control System for the SPS", CERN 75–20

[2] M.C. Crowley–Milling, "Experience with the Control System for the SPS", CERN 78–09

[3[R. Rausch, "Real Time Control Networks for the LEP and SPS Accelerators", in *Control Systems for Experimental Physics*, Villars, Switzerland, September 1987, pp. 244–247.

P.G.Innocenti, "The LEP Control System", in *Accelerator and Large Experimental Physics Control Systems*, Vancouver, Canada, 1989, pp 1–5

P. Lienard, 'The SPS and LEP Control Network Architaecture", in *Accelerator and Large Experimental Physics Control Systems*, Vancouver, Canada, 1989, pp 215–220.

[4] R. Bailey, J. Ulander, I.T. Wilkie, 'The New Applications Software for the CERN SPS", in *Control Systems for Experimental Physics*, Villars, Switzerland, September 1987, pp. 158–163

[5] A. Ogle, J. Ulander, I.T. Wilkie, "Experience with Workstations for Accelerator Control at the CERN SPS", in *Accelerator and Large Experimental Physics Control Systems*, Vancouver, Canada, 1989, pp 437–441.

[6] M. Tyrell, "The LEP Alarm System", these proceedings.

Experience Controlling the LAMPF-PSR Accelerator Complex*

S. Schaller, R. Stuewe, E. Bjorklund, M. Burns, T. Callaway, G. Carr,
S. Cohen, M. Harrington, D. Kubicek, R. Poore, and D. Schultz
Los Alamos National Laboratory
Los Alamos, New Mexico, 87545, USA

Abstract

In recent years, control system efforts at LAMPF have emphasized the provision of uniform control for the LAMPF linear accelerator and associated beam lines and the Proton Storage Ring and its associated beam lines. The situation is complicated by the presence of several control philosophies in the operator interfaces, data base mechanisms, and front end data acquisition and control interfaces. This paper describes the current system configuration, including the distributed operator interfaces, the data and control sharing between systems, and the use of common accelerator diagnostic software tools. Successes as well as deficiencies of the present system will be discussed with an eye toward future developments. *

I. BACKGROUND

The Clinton P. Anderson Meson Physics Facility -- also known as LAMPF, the Los Alamos Meson Physics Facility -- is composed of an 800 MeV proton linac plus associated beam lines and targets, and an 800 MeV Proton Storage Ring (PSR) plus its beam lines and a neutron spallation target that serves the Los Alamos Neutron Scattering Center (LANSCE). The linac accelerates beams of H+, H-, and polarized H- (referred to as P-) ions up to 120 times per second in pulses of up to 1000 microseconds width. The average H+ beam current can be as much as 1 mA. The proton storage ring serves as a beam compressor, taking a full H- macro-pulse from the linac and ejecting it in several hundred nanoseconds.

When LAMPF was built in the 1960's it was one of the first accelerators to be designed for computerized control. Since the IEEE CAMAC standard did not exist at that time, a significant amount of effort went into the design and construction of LAMPF-specific data acquisition hardware. The system that resulted was called RICE (Remote Instrumentation and Control Equipment). RICE hardware is still used for more that 60% of the linac equipment. Over the years, the control system was expanded to include CAMAC hardware accessed on demand through remote computers. With the addition of remote operator consoles, the initial star architecture has evolved into a much more distributed configuration.

The Proton Storage Ring was designed in the early 1980's to be independent of LAMPF with a separate control room and beam lines. As a consequence, the PSR Control System was designed and implemented with only minimal consideration of LAMPF requirements. The PSR system did provide recognition of its effect on linac timing requirements and it used a device naming scheme that was similar to LAMPF's. The PSR system emphasized continuous update of a centralized database.

In 1988, responsibility for the Proton Storage Ring was transferred to LAMPF. This paper describes the present configuration of the two control systems and the attempts that have been made to integrate them in a useful manner. We conclude with a brief description of our plans for the future. More information about our plans can be found in a companion paper at this conference [1].

II. CURRENT CONFIGURATION

A. LAMPF Control System (LCS)

The evolution of the LAMPF Control System (LCS) has been described in detail elsewhere [2-4]. The LCS is currently composed of a network of VAX computers connected via an Ethernet using DECnet for communications. Computer systems in the LCS network are of two types, (Figure 1). A typical LCS operator console computer runs VMS and drives one or more LCS operator consoles. Such a computer may also have a CAMAC-based data acquisition and control capacity. A typical LCS data acquisition front-end computer runs the VAXELN real-time kernel and handles hard real-time data acquisition through CAMAC. The VAXELN nodes do not have local disks.

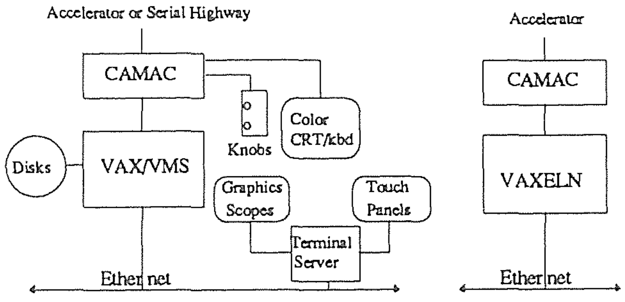


Figure 1. LAMPF Operator Interface and Front End Computers

Each LCS operator console is composed of one or more color character-cell CRTs which are shared between a number of application programs, several graphics scopes, trackball-based touch panels, and a set of analog control knobs. The graphics scopes and touch panels are attached to the computer

---

* Work supported by the U.S. Department of Energy

through terminal servers. The color CRT and knobs are attached through CAMAC.

The color CRT in the LCS operator interface allows any LCS program to be called up at the operator's demand. This interface also gives the operator access to a number of supervisory tools which allow the state of any devices to be displayed and controlled. The touch panels also allow access to a fixed number of application programs.

LCS operator consoles are now supported in the LAMPF Central Control Room (CCR), the Injector Control Room (ICR), and the LANSCE Control Room (LCR). (See Figure 4 for a geographic representation of this distributed functionality.) The main CCR control computer was the center of the original star configuration. It still maintains its central position as it drives four of the five LCS operator consoles in CCR and serves as the central repository for LCS software and databases.

Since the RICE hardware was and still is a primary feature of the LAMPF Control System, a VAXELN front-end computer is dedicated as its interface with the rest of the control system (Figure 2). The RICE system is composed of 73 hardware data acquisition modules arrayed along the linac and in the injector and experimental areas. A distinct advantage of the RICE system is that it supports simultaneous timed data takes on each RICE modules. This provides a very powerful method for acquiring longitudinal snapshots of the entire linac at a particular time on a particular beam pulse. For untimed data takes, data caching facility is provided. In addition, the RICE system interfaces with the accelerator "fast protect" system. If a hardware monitor determines that too much beam is being spilled, the fast protect hardware sends a signal that simultaneously inhibits the injector and notifies the RIU computer that a fast protect has occurred. The RIU computer immediately reads the state of all hardware monitors to determine where the fast protect occurred.

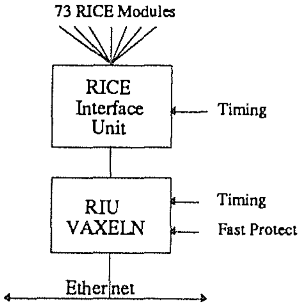


Figure 2. LAMPF RICE Interface Front End Computer

LCS data acquisition is demand driven. Each node in the LCS network contains a static database derived from a master database on the main CCR computer. Application programs request data and issue commands through a standard data access interface which uses the local database to resolve device addresses at run-time. Application programs may be split up among several nodes. A locally designed Remote Procedure Call (RPC) interface allows the pieces to communicate without dealing with the complications of DECnet. Programs that know they need large amounts of data can improve their system throughput by forming "aggregate devices." The LCS data access interface makes use of information supplied by the program to optimize network usage.

Because of the uniformity of data acquisition interfaces, if the correct application programs and databases are supplied to it, any LCS operator console on the network could run the entire accelerator.

B. PSR Control System

The PSR Control System attempts to achieve high data throughput and reasonable operator interface response by tightly coupling a central database to external computers that are continuously polling data. Detailed descriptions of this system have been published elsewhere [5-6].

A diagram of the PSR operator interface and front end computers is given in Figure 3. All PSR operator consoles are attached to the PSR VAX. This machine serves as the operator interface computer and the central data concentrator. A typical PSR operator interface screen consists of a color graphics scope whose face is overlaid with a touch sensitive surface. The graphics scope is driven directly from the computer bus; the touch panel is driven through a terminal server on Ethernet. A set of analog control knobs is controlled via CAMAC. The top level screen on each color graphics scope provides an entry to a tree-structures menu of possible programs that can be started.

The PSR front-end computers are PDP-11s each connected to a CAMAC serial highway for data acquisition and control. These front-ends are know as Instrumentation Sub-Systems (ISSes). They continuously update their local databases with data from their serial highways. The ISSes are also connected to each other and the PSR VAX via a separate CAMAC serial highway over which the PSR VAX reads the latest data from the ISS databases and transfers changes in control values.

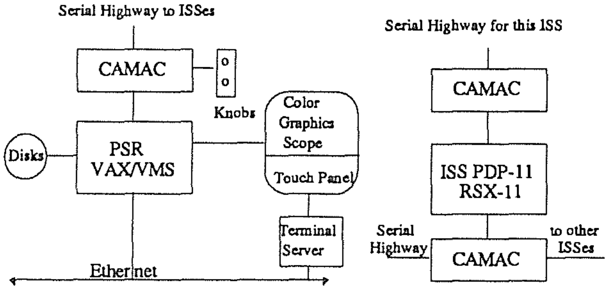


Figure 3. PSR Operator Interface and Front End Computers

Application programs running on the PSR VAX typically have exclusive access to a single graphics scope. The programs access data and set control values in the central database. They can also be notified asynchronously of changes in database values. Device data addresses in the central database are resolved at link time for many of the PSR application programs. This limits run-time overhead, but results in problems if the database structure changes.

C. Control Systems Integration

When the Proton Storage Ring was first commissioned, it was controlled entirely from LCR. LAMPF at that time had operator consoles in CCR and ICR, although most operations activities took place in CCR.

The first attempt at LCS/PSR integration was to place PSR control consoles in CCR. Since we also wished to keep LCR available for PSR beam development, we had to pull cables between the two control rooms to physically connect the remote PSR consoles with the PSR VAX. At the same time a LCS console (CPU, color CRT, graphics scope, and knobs) was installed in LCR.

We then approached the more difficult job of sharing data and controls between the two systems [7]. By adding software (mainly run-time libraries and a copy of the LCS database) and LCS console hardware, we were able to make the PSR VAX into an LCS operator interface computer. This meant that PSR application programs could access LCS data through the LCS data interface. Several PSR applications that were interested in linac data were so modified.

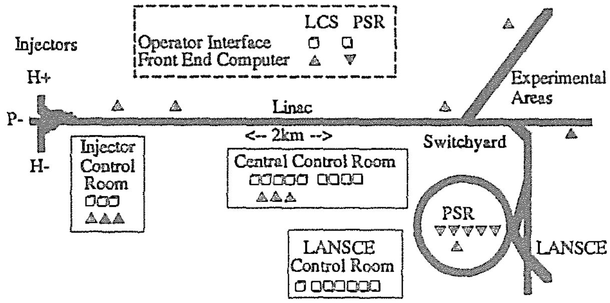


Figure 4. Distributed Functionality in the LAMPF-PSR Network

The situation with LCS programs was a bit more difficult. Since only programs residing on the PSR VAX could access the PSR database, we added an LCS Data System server process to the PSR VAX. This process (which is a variant on the standard LCS Data System server) handles network requests for information on PSR devices. Since the LCS programs make standard requests for data, all the LCS programs, including the supervisory display and control tool, the LCS knobs, and the LCS data archiver could get PSR data.

Figure 4 show the geographic distribution of LCS and PSR consoles and front-end computers. This figure does not show details of CAMAC highways, Ethernet connections, RICE cabling, or timing distribution.

III. EXPERIENCE

A. Front Ends

The RICE hardware system presents some unique problems to the control system. Since more than 60% of the LCS hardware is attached through the RICE system and is only accessible through the RIU front-end computer, this represents a major bottleneck. The fact that we have a limited amount of RICE hardware available means that we cannot easily expand the system.

The RICE interface can only perform one timed data take per beam pulse. This severely restricts tuning operations which are typically performed at low rep rates. We would like to improve on this performance even though we would like to keep the capability for doing synchronized timed data takes. Other RICE problems include not being able to send analog commands to more than one device in a RICE module at a time. This constraint creates problems when one is trying to scan wires in two wire scanners in the same RICE module.

The primary problem that we have found with the PSR ISSes is that of performance. The PSR equipment modules which reside in the ISSes are flexible and can acquire data rapidly. But the overall response is slow because data is scanned and transmitted regardless of its usefulness. The inherent flexibility becomes a problem because some modules respond differently to the standard read and command requests that are issued from the application programs. We need a carefully designed application program model of the world and disciplined equipment module implementations that ensure standard responses.

B. Databases

A database system in an experimental installation must be able to add and change device definitions quickly without taking the control systems offline. The LCS database succeeds in this respect; the PSR database does not. After a PSR device definition has been modified, it can take hours to regenerate a new PSR database. To be able to communicate with PSR devices, the LCS database must contain the PSR device names. This is possible because both systems use the same device naming scheme. Unfortunately, there is no automatic scheme for rationalizing the names that occur in the two databases. For now we use editors to compare lists of names to determine what should be a changed.

C. Application Programs

The difference in design philosophies between the two control systems is most noticeable in the application programs. The absence of a notification on change in the LCS data access interface makes it hard to allow PSR programs to access LCS data through the PSR data access mechanism. As a compromise, we have made it possible for PSR programs to access LCS data through the LCS access mechanism. On the other hand, it was relatively easy to enable LCS programs to access (possibly old) PSR data. For application programs driving a future common LCS/PSR operator interface we should like to have data from both systems provided in a uniform manner.

D. Operator Interfaces

Neither the LCS nor the PSR operator interface lend themselves to upgrade. The LCS technology is old and cannot integrate graphics and control functions. The PSR screens are becoming unmaintainable and cannot be moved to other nodes in the network because PSR applications need to access the central PSR database directly. The LCS supervisory tools allow operators to directly access any device. The PSR tools must be recompiled to allow access to new devices. We have found that the flexibility provided by the LCS interface is vital in running a basically experimental accelerator.

E. Reliability and Maintainability

Hardware reliability and maintainability has been a key issue during recent accelerator runs. The RICE hardware is getting old and becoming difficult to maintain. Replacement hardware is no longer being manufactured. While CAMAC hardware is available for replacements and additions, its use in harsh environments sometimes leads to short lifetimes.

The use of long serial CAMAC highways, especially in the PSR system, has led to difficulties in problem isolation. Frequently it is necessary to remove one crate controller/fiber optic driver at a time from the highway in order to isolate a fault. This can be a very time consuming operation, especially if it has to done during production.

We have also been concerned about single points of failure within the control system. As the systems stand now, the failure of a single LCS operator interface or front-end computer only means that the CAMAC attached to that machine is inaccessible. As mentioned above, with the correct data files and programs, any LCS operator interface computer in the network, including the PSR VAX, can run the accelerator.

The failure of the RIU front-end computer would be more serious for then we would loose all RICE data, a significant portion of the accelerator's data.

Loss of the PSR VAX would mean loss of the entire PSR system since the serial highway connections are only to that machine and the central PSR database resides on it.

IV. THE FUTURE

The concerns described in the previous sections are being dealt with through several projects being currently planned or implemented at LAMPF. The projects and several other considerations are described in detail in a companion paper at this conference [1]. In the remainder of this paper we will briefly summarize these projects.

At the lowest level, the front end data acquisition hardware will be upgraded to meet new requirements for reliability and maintainability. The plan is to use VAXELN-based micro-VAXes as the standard front end computer. These front ends will be used to replace the PDP-11s currently being used for the PSR ISS interface to CAMAC. At the same time, it is planned to replace the RICE hardware with CAMAC and again use VAXELN front ends. The effect of these two projects will be to unify all device access through a common client-server model.

A new common operator interface project has also been proposed. A common interface will reduce both training and maintenance requirements. We plan to use VAX-based workstations as the primary operator interface. We plan to use a user interface management system to keep the development and maintenance of the operator interface more manageable.

Since the new data access mechanism will not automatically put the data in the PSR database, existing PSR application programs may have to be changed. There is the possibility that some of these programs may be rewritten to make use of the new common operator interface. In the long run, this is what we hope to do with all application programs.

To improve the overall responsiveness of the control system, we hope to pursue several hardware upgrades beyond replacing the front end computers and introducing operator interface computers. Since, for a while, some application programs will still be using the old interfaces through the LCS and PSR central control computers, we hope to replace them with higher performance VAXes which can be clustered to provide hardware and file backup for each other.

With these upgrades we will be able to respond to increased demands on the LAMPF control systems in the future. Of most immediate interest are the controls necessary to support the proposed Pion Linear Accelerator (PILAC) to be built at one of LAMPF's beam lines. There are also proposals to upgrade the PSR from 100 to 300 μA to drive a possible pulsed lepton source, and to use LAMPF for prototype work in using an accelerator to transmute radioactive waste.

V. REFERENCES

[1] R. Stuewe, S. Schaller, et. al., "Future Directions in Controlling the LAMPF-PSR Accelerator Complex," these proceedings.
[2] G. Carr, S. Schaller, et al., "The Status of the LAMPF Control System Upgrade," Proc. Europhysics Conference on Control Systems for Experimental Physics, CERN Yellow Report 90-08, (1990) p.107.
[3] S. Schaller and E. Bjorklund, "Distributed Data Access in the LAMPF Control System," Proc. 1987 IEEE Particle Accelerator Conf., Washington, DC (IEEE Publishing, New York, 1987) p. 745
[4] S. Brown, S. Schaller, et al., Proc. 2nd Int. Workshop on Accelerator Control systems (North-Holland, 1986) p. 122.
[5] P. Clout, et al., ibid., 1986, p. 116.
[6] P. Clount et al., "The Proton Storage Ring Control System," IEEE Trans. Nucl. Sci. NS-30 (1983) p. 2305.
[7] S. Schaller, "Providing Common Data Access for the LAMPF and PSR Control Systems," Nucl. Instr. and Meth., A293 (1990) p. 416.

STATUS REPORT ON THE ADVANDED LIGHT SOURCE CONTROL SYSTEM

S. Magyary, M. Chin, M. Fahmie, H. Lancaster, P. Molinari, A. Robb, C. Timossi and J. Young

Accelerator and Fusion Research Division
Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720
U.S.A.

Abstract

This paper is a status report on the ADVANCED LIGHT SOURCE (ALS) control system. The current status, performance data, and future plans will be discussed. Manpower, scheduling, and costs issues are addressed.

I. INTRODUCTION

The ALS control system was designed around the concepts of parallel processing, high CPU and I/O bandwidth, and human-friendly interface. Figure 1 shows the system architecture and its five primary layers (for details of the system see References [1] and [2]). Layer 1, represented by the Intelligent Local Controllers (ILCs), interfaces to the accelerator hardware and communicates with Layer 2, the Collector Micro Module (CMM). Layer 3 is the Display Micro Modules (DMM) that has bus access to the CMM and in turn communicates with the operator stations (Layer 4) via serial links. The operator stations are high-performance Personal Computers that have Ethernet network (Layer 5) access to file servers and other network services.

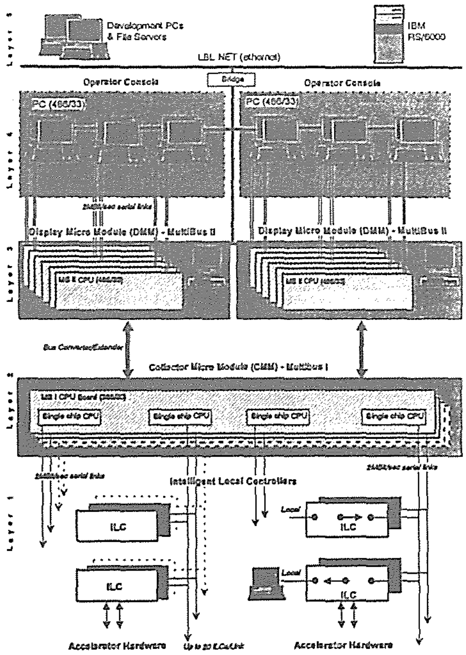


Figure 1. ALS control system architecture.

The ALS consists of an Electron Gun, a Linac, a Linac-to-Booster line, a Booster, a Booster-to-Storage-Ring line, a Storage Ring (SR), and a number of user beamlines. The control system is currently operating the existing parts of the ALS accelerator hardware consisting of the Gun, Linac, Linac-to-Booster line, and the Booster; the Booster-to-Storage-Ring line is being implemented now. The Storage Ring accelerator hardware is under construction; completion is expected sometime during the second quarter of 1992. We will then be ready to begin commissioning the SR via the control system, both locally and from the Control Room.

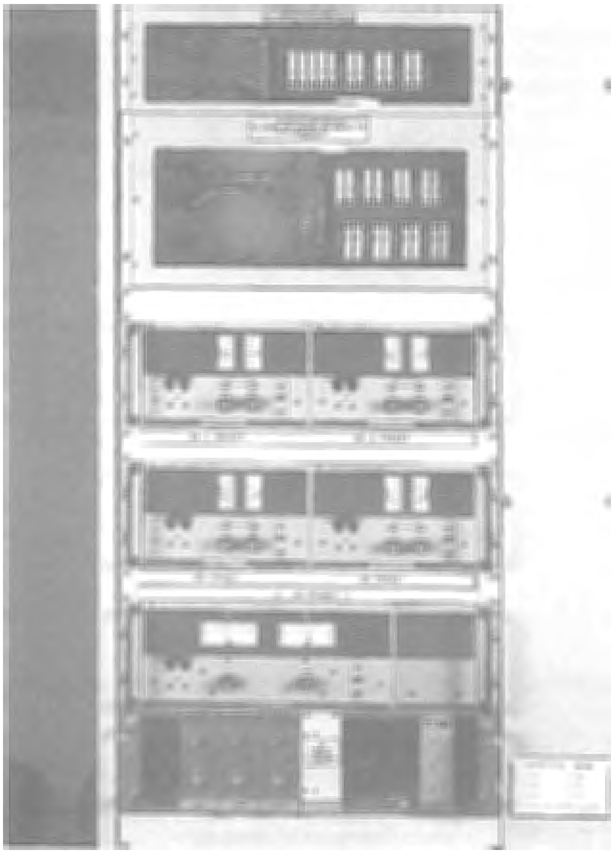


Figure 2. Typical ILC installation.

II. LAYER 1 (INTELLIGENT LOCAL CONTROLLERS)

The ILC is an intelligent controller consisting of an 80C186 main processor, an 80C187 math co-processor, and an 80C152 serial-control processor sharing 64 Kbytes of battery backed memory. In addition, it has on board I/O resources of four 16-bit DACs, four 13-bit ADCs, 24 bits of digital control, and an SBX bus for expansion. It is a low power (< 5 watts), 3U high Eurocard-based controller in a shielded metal can that can communicate at a 2 Mbit/sec rate using twisted pair cabling. We had commercial companies build 200 of these first generation ILCs. Twenty of them (10%) were not functioning when received from the manufacturer; ten had minor problems (chip leads bent, missing chips, infant mortality, etc.) and were repairable. The remaining ten we have not attempted to repair (they have missing traces or shorts on the circuit boards) since we decided the cost of repair was not justified. We are currently using about 140 ILCs in the accelerator, with an additional 20 to be used for the Booster-to-Storage-Ring line. See Figure 2 for a typical installation showing 3 ILCs, Opto-22 interface, and the 3 quadrupole power supplies. We expect to use an additional 500 ILCs to complete the project (Storage Ring, Beamline Front Ends), however, these will be the next generation design. These new ILCs will have 16 Mhz, 80C186EB chip as the main processor (a 60% speed improvement over the older ILCs' CPU, and also lower-power dissipation), 256 Kbytes of memory, 16-bit ADCs and DACs, 16-bit SBX interface, a serial channel, and 28 Boolean lines. The analog and digital design is completed; layout and prototyping/testing will start shortly. The cost of the current ILCs is $650 each; the new ones will be about $950. To exercise the ILCs, and test the accelerator hardware, we use laptop computers (80386/80486-based), connected directly to the ILCs, using much of the same software as we use on the operating stations in the Control Room.

Figure 3. DMM, CMM, fiber optic interface, and file server.

III. LAYER 2. (COLLECTOR MICRO MODULE)

The CMM (Figure 3) contains all the data gathered from all the ILCs (i.e., it represents the entire accelerator database at any moment). The ILCs are connected to the CMM via fiber-optic lines. The serial communication on these lines is bi-directional (though using only a single fiber), and the bit rate is 2 Mbit/sec. We are currently using 12 of these lines for a total I/O physical bandwidth of 24 Mbit/sec. We use a commercial (Intel) 20 MHz, 386-based Multibus I board with 4 single chip processors (via custom built SBX modules) to service 4 serial lines. Therefore, to service the 12 lines currently in use, 3 Multibus I processors are required. These boards allow us to handle approximately 800 messages, of about 75 bytes each, per second per line. Therefore, the total I/O bandwidth currently coming into the CMM is approximately 720 Kbytes/sec (12 x 800 x 75). This represents, on the average, about a 10 refresh/sec of the entire active part of the current accelerator database. We are currently evaluating a 33 MHz, 486 Multibus I board and are designing a new SBX interface that would allow us to service 8 lines per Multibus board at about 1600 messages/line/sec. At project completion, we plan to have approximately 64 lines (using 8 Multibus I processors operating in parallel) for a total I/O bandwidth of 128 Mbits/sec, and a useful data rate 7.5 MBytes/sec. This would represent, on the average, about a 15 to 20 per second refresh rate of the active database of the entire accelerator. We are also exploring the possibility of a second CMM; this would double our I/O bandwidth to 240 Mbit/sec at a modest cost.

IV. LAYER 3. (DISPLAY MICRO MODULE)

The DMM (Figure 3) consists of a 20-slot Multibus II System, bought as a unit from Intel Corp., that has fast parallel access (via a bus converter) to the CMM. This system contains a SCSI disc controller, an Ethernet interface, and a number of high performance singleboard computers. Under current operation, we are using 2 DMMs (we had not planned to install the second DMM till Storage Ring commissioning, so we are ahead of schedule in this area) to access the database in the CMM. The DMMs currently use RMX II as the real-time operating system (we are evaluating using RMX III, a full 32-bit system) using standard Intel hardware and software. The first DMM currently supports 6 (we have tried 7) commercial (Intel) 25 MHz, 486-cpu boards, while the second DMM currently uses 3 CPUs. The CPUs within each DMM operate in parallel, and each has 2 serial links (we promised 1, so this doubles I/O performance in this area) that directly connects it to the operator station. Each of these links has the same configuration as the links between the ILCs and the CMM (i.e., 2 Mbit/sec). At project completion, we will support at least 6 operator stations per DMM, for a total of 12 or more for the whole accelerator. We have enough bus bandwidth in the CMM so we could support even a third DMM. We plan to upgrade the CPUs in the DMM with processors that will be at least 2.5 times a fast as the current ones.

V. LAYER 4. (OPERATOR STATIONS)

The operator station is the human interface of the Control System; it presents accelerator data, including real-time data, scope traces, and live video (via multi-media programs and hardware) to the operator. The operator can use mice, keyboards, or nine dynamically assignable/labeled knobs. Six operator stations make up a console (Figure 4), and each console is supported by one DMM. The operator stations use 33 MHz, 486-based AST Personal Computers (PC), but they are upgradeable to faster CPUs via a simple card swap. Most use Windows 3.0 as the operating system, a few use OS/2 1.2. The serial communication links to the DMM allow us about 800-1000 database accesses/sec for each PC in the database client/server "request" mode. In the "driven" (i.e., when the DMM drives the PC) we can achieve about 1500 messages/sec. Both of these rates are CPU limited, we expect them to be about 2500 and 3000 messages/sec (at those rates we will be I/O limited) respectively at project completion. Since we plan to have at least 12 operator stations, we will attain accesses in excess of 30,000/sec; however, even at these rates, we are using only about 10% of the bus and data bandwidth available in the CMM-DMM combination. These facts demonstrate the very high performance available in our star-based, shared memory approach to control system design. The choice of the PC as our human interface has proved to be a judicious one. As the popularity of Windows grows, the availability of commercial software for use in the control system grows rapidly. We currently make extensive use of DESIGNER (graphic editing tool), EXCEL (spread-sheet), TOOLBOOK (HyperCard-like package), VISUAL BASIC , TURBO PASCAL for WINDOWS, as well as the usual languages C, C++, etc. With these PCs, we also have available the usual collection of word processors, databases, and utilities (screen capture, etc.). The use of this commercial software has greatly increased our productivity, and has allowed us to keep our staffing requirements very low. The one area where we need improvement is in support of full 32-bit modeling applications. These are currently done on workstations and can access the database via Remote Procedure Calls (RPC). We plan to shift this work to OS/2-2.0, or its equivalent, as soon as possible. In the meantime, X-Windows-based applications can appear as a window on the PCs under Windows 3.0 or OS/2, using commercial software.

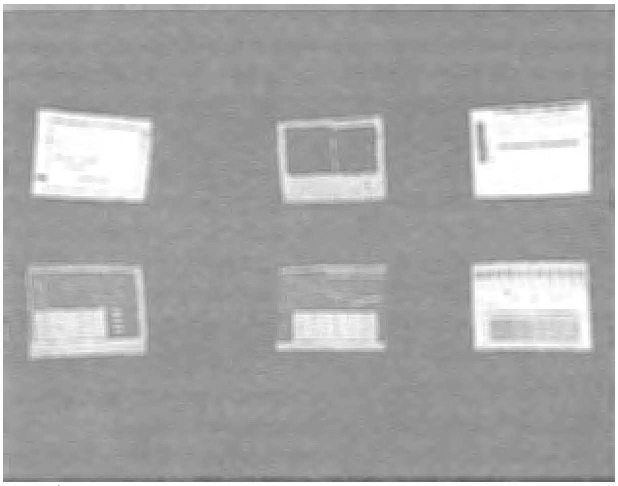


Figure 4. Operator console with six operator stations.

VI. LAYER 5. (NETWORK)

The PCs are networked via Ethernet to a Laboratory-wide network. This allows workstations, etc., to access the database via RPCs. An IBM RS6000 Workstation is used by the physicists for modeling and 32-bit numeric intensive applications. We also have a PC on this network with a 600-Mbyte disk as a file server for the operator stations and software development. A spare server is available to minimize down-time in case of failure by the main server. This network will also be the "user" interface into the control system for wiggler/undulator and beamline front end controls. A protection scheme to limit user control, to specified devices only, is under development.

A. Scheduling and Costs.

The control system is being used to commission the accelerator as it is being built. We are on target both in terms of cost and schedules, and are beginning to shift over to pre-operation. The staffing to date has been exactly as projected (with five programmers, one coordinator, and one half of an electronic designer) over the length of the construction project. Cost to date is approximately $3.65M of a total projected cost of $5M. Storage Ring commissioning is scheduled for the 2nd quarter of 1992, with project completion in 1993.

VII. ACKNOWLEDGMENT.

This work was supported by the Director, Office of Energy Research, Office of Basic Energy Sciences, Materials Sciences Division of the U.S. Department of Energy, under Contract No. DE-AC03-76SF00098

VIII. REFERENCES.

[1] Magyary et al., "Advanced Light Source Control System," Lawrence Berkeley Laboratory, LBL 26028 and Proc. 1989 IEEE Particle Accelerator Conf., Chicago (IEEE Publishing, New York, 1989).
[2] Magyary et al., "The Advanced Light Source Control System," Nuclear Instruments and Methods in Physics Research, A293 (1990) 36-43, North Holland.

LESSONS FROM THE SLC FOR FUTURE LC CONTROL SYSTEMS*

Rusty Humphrey

Stanford Linear Accelerator Center, Stanford University, Stanford, CA 94309

The SLC control system is the dynamic result of a number of forces. The most obvious force is the functional requirements of the SLC itself, but other forces are history, budget, people, available technology, etc. The plan of this paper is to describe the critical functional requirements of the SLC which caused significant development of the control system. I have tried to focus on functional requirements as a driver, and I will describe some solutions which we have implemented to satisfy those requirements.

The important functional requirements drivers for the control system discussed in this paper are:

- ➪ Repetition rate
- ➪ Sensitivity to orbit distortion
- ➪ Stability/Automation
- ➪ Accelerator Development

REPETITION RATE

The SLC runs for physics production at 60 or 120 Hz. At 120 Hz, 5×10^{10} particles per bunch, 3 bunches/beam pulse, and 50 GeV, the average power is 150 kW. If the beam has a small enough cross sectional area, such a beam has caused damage to beam vacuum pipes, beam vacuum flanges, collimators, or other beam line components by heating. Such events occur because the beam has become "errant"; that is, it has wandered from its nominal orbit, and is actually striking the device. If this situation is not detected, then more and more energy is put into the device, as the SLC pulses keep coming. The first issue is to detect the event, and turn off the beam. There are a number of classic methods of such detection (ion chambers, beam current comparators, etc.), and the SLC uses them.

Once the event is detected, how does one fix the problem? Usually the answer is to steer or tune the machine. But now a situation, which appears as a form of "relaxation oscillator," happens. To tune the beam, one needs beam in the machine. But because the beam is mistuned, the machine protection system detects the same problem again and turns off the beam again. How does one break this impasse?

The first, and obvious answer is to tune at a lower beam intensity; instead of running with 5×10^{10} particles, tune with 2×10^{10}. This doesn't work in general. The SLC with 2×10^{10} particles is a sufficiently different machine from the SLC with 5×10^{10} particles that the problem often disappears at 2×10^{10}, only to reappear when the current is raised to 5×10^{10}.

The next answer is to tune at the same beam pulse intensity, but to lower the repetition rate. This is, in fact the technique that is used at the SLC. However, it does not work to simply lower the repetition rate of all components in the machine to 10 or even 1 Hz. Power is dissipated in the rf and pulsed magnet systems, and lowering the repetition rate in such components changes their characteristics. Therefore, an effective rate limiting strategy requires that the rate of running the pulsed components of the machine not be changed, but that only the injection of electrons and positrons be moved to the lower rate.

The above discussion is an overview of the simplest situation; and even it isn't really simple—how the creation and injection of positrons is handled is problematic even in this situation. More complicated scenarios are also possible in the SLC [1].

Another issue for the Machine Protection System is configuration flexibility. As the SLC configuration is changed during tuning or machine studies, the requirements on machine protection change. An obvious example is a repetition rate change from 60 to 120 Hz. A less obvious example is changing the place where the beam is stopped. It is a requirement of the machine protection system that it react to such configuration changes in as seamless a manner and as prompt as possible. At the SLC, this functionality is provided by means of the timing system, which includes distribution of timing "patterns" which allow pulse to pulse timing configuration changes. This functionality is being augmented because it is required by a project to upgrade our present Machine Protection System [2], and because it is needed for the next phase of our Fast Feedback system.

To summarize the functional requirements: The repetition rate for a linear collider can allow errant beam to damage or destroy beam line components. A protection scheme is required which detects such situations, which limits the beam, and which allows retuning of the machine to stop the situation. It is required that retuning be done at or near the beam conditions which cause the errant beam. In addition, the machine and its machine protection system must be easily and quickly reconfigurable.

---

* Work supported by Department of Energy contract DE-AC03-76SF00515.

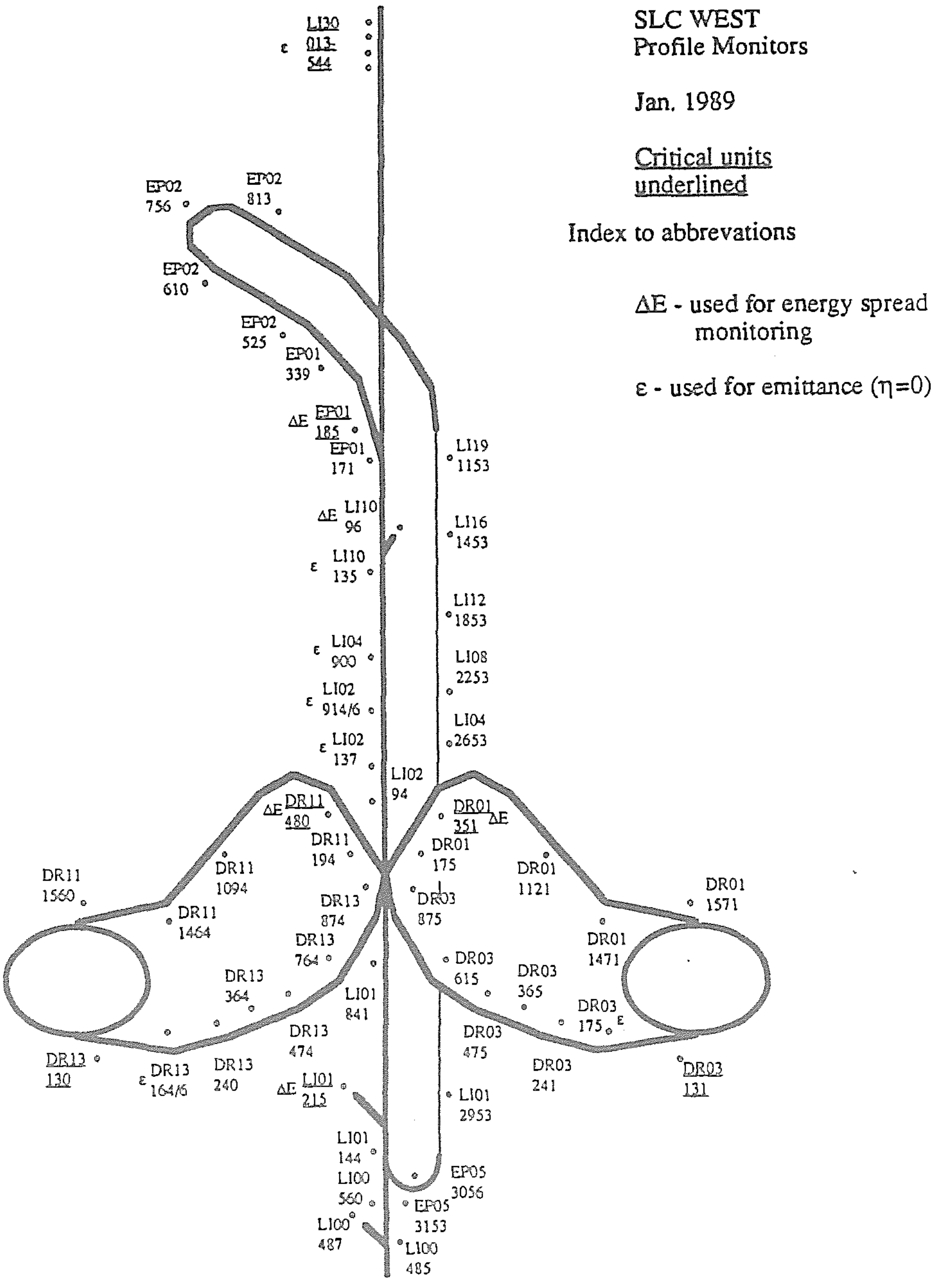


Figure 1. Location of beam profile monitors in the SLC injector, damping rings, linac and positron production systems.

SENSITIVITY TO ORBIT DISTORTION

In the SLC, emittance and other parameters of the beam are affected by orbit distortions. One easy way to understand this is to remember that wake field tails are caused by off axis beams in the linac's disk loaded wave guide. As a result of this sensitivity, the mix of beam diagnostic systems required for the SLC is affected. Diagnostics which measure beam shape, beam size, and emittance are many. As shown in Figures 1 and 2, there are approximately 100 beam profile monitors and 37 wire scanners.

The beam profile monitor system has been described elsewhere [3]. As noted there and elsewhere [4], the use of profile monitors is destructive to the beam, but they allow shape changes to be observed in real time and give detailed information of transverse tail formation. (See Figure 3.) In concert with an adjustable upstream quadrupole, beam profile monitors can be used to measure emittance [5].

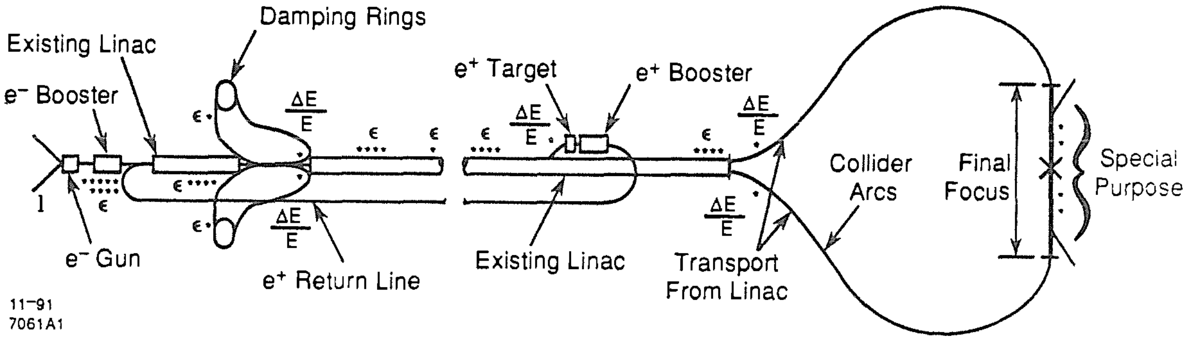


Figure 2. Location of wire scanners in the SLC.

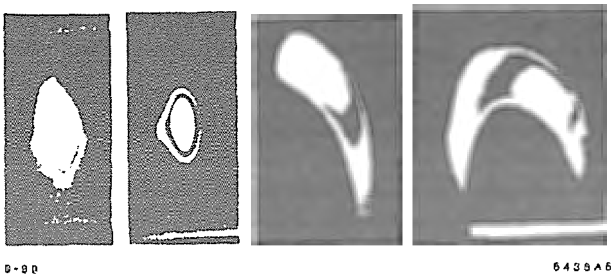


Figure 3. Images of an electron bunch on a profile monitor at 47 GeV showing wakefield growth with increasing oscillation amplitudes. The images from left to right are for a well-steered beam, a 0.2 mm oscillation, 0.5 mm oscillation and a 1.0 mm oscillation, respectively. The beam intensity is 2×10^{10} electrons. The core sizes σ_x and σ_y are about 120 mm.

The wire scanners have been discussed elsewhere [6]. The beauty of wire scanners is that they allow nondestructive measurement of the beam emittance, and thus could be used as an online device in, for example, beam feedback systems (we have not yet done so).

John Seeman has pointed out the need for what he calls "corroborating measurements." As an example of what this term means, consider the fact that emittance can be measured by both profile monitors and wire scanners. The presence of two techniques allows the results of such measurements to be compared. If the measurements are equivalent, then they corroborate (or confirm) one another. This increases the credibility of the results—an important factor in a prototype accelerator.

Beam position monitors (BPMs) used in the SLC number approximately 1700. All the BPMs in the linac itself are instrumented for single pulse data acquisition; every BPM so instrumented can be read out, under control of the timing system, on any given pulse for a particular beam bunch. BPM systems in the SLC arcs and in the damping rings have multiple BPMs which are multiplexed into a common data acquisition module; this precludes reading all the BPM inputs into one of these modules on the same beam pulse. However, over the past year, we have had a couple of projects to "demultiplex" BPMs; that is, to instrument more BPMs in the same way as the linac BPMs so that orbit measures on a single beam pulse can be done. The builders of future linear colliders need to look carefully at the requirements for single pulse orbit measurement.

The impact of these beam diagnostic systems on the control system is large. Fundamentally, the data acquisition requirements for a linear collider correspond to that of the "first turn" for a circular collider. The ability to take a single pulse "snapshot" of the orbit, or a snapshot of many parameters associated with the beam or with individual pulsed devices is a requirement. As the references detail, emittance and beam shape measurements require sophisticated image processing and accelerator matrix manipulation and fitting. As the maps of profile monitors and wire scanners show, and as the number of BPMs implies, these systems are everywhere, and time spent on generalization and sophistication is well spent.

To restate the functional requirement: linear collider operation requires careful attention to diagnostics which measure beam orbit position and distortion, emittance, and beam shape.

STABILITY, AUTOMATION

The SLC is a large complicated device. Stability of the SLC is a large problem. Feedback systems, in which the control computer system is an active component of the feedback loop, have been operational at the SLC since 1988. Feedback based on signals derived from beam diagnostic instrumentation allows a much higher degree of control over the beams, since these data can be acquired from many sources and statistically fit. Single device tolerances could never provide this level of stability. The main application of these feedback systems is steering (launch angle and position); but feedback systems to correct energy, energy spread, and collision point are also used.

The earliest version of these was "slow feedback," with update times measured in tens of seconds; such loops are closed through the VAX mainframe which is the highest hierarchical level in the SLC control system. This was quickly augmented by prototype pulse-to-pulse feedback ("fast feedback") systems using a dedicated microprocessor based system, instrumentation, and controlled steering supplies. This prototype system was a very successful, but could only be replicated with difficulty and was difficult to maintain. We have since generalized this prototype and integrated it into the SLC control system. That generalization is propagating at a rapid rate to a large number of installations in the SLC, replacing both the prototype version of itself as well as many of the older "slow feedback" applications. This system is described in another paper being presented to this conference [7].

One of the major benefits of these fast feedback systems is the step forward in automation that they allow for accelerator operations. As described elsewhere [8], the SLC control system logs a number of different events on a continuing basis. One such class of events logged is "knob turns"; i.e., each time an operator turns a software-defined knob, that event is logged. As a result, we know that fast feedback has decreased the required intervention of operators to do knob turns by as much as 80%; fast feedback is doing the knob turning for us.

ACCELERATOR DEVELOPMENT

The SLC is the prototype for a linear collider. The SLAC staff is working to understand how a linear collider works. One of the SLC accelerator physicists has noted that "...there are more interesting accelerator physics tests being proposed each day than there is accelerator time to perform them" [4]. The environment is such that there are numerous questions to be answered and there is often the need find answers *quickly* so that the answers can be incorporated into operation. It is an essential functional requirement that the control system supply tools that allow the staff to do machine physics experiments which have never before been considered.

The major tool—actually a set of tools—for this is the Correlation Plot Facility, described in a poster session paper of this conference [9]. This powerful software provides a set of tools for realtime online analysis, fitting, plotting, control and measurement of a large number of variables. The facility is well integrated into the SLC control system, and programs or functions which are developed for physics studies are often incorporated into operational software [10].

This functional requirement will exist for the next linear collider, since it will be built on an experience base of one—the SLC.

COMMENTS

The control system for an accelerator must satisfy many functional requirements—many more than the four described above. These four were described because SLAC's experience shows that they are, in some way, unique to the class of linear colliders.

There are other functional requirements which are common to all accelerators. And there are functional requirements which are unique to the SLC—a prototype linear collider based on the existing SLAC linac. Neither of these classes of functional requirements have been discussed, although some of the solutions described above help to meet them.

The four functional requirements described above have been a challenge which has been met by a large team of highly committed people. Some of that team is named in the references, but there are many, many more. I would like to thank Marty Breidenbach, Ewan Paterson, Nan Phinney, Marc Ross, John Seeman, and John Sheppard for recent discussions on this topic.

REFERENCES

[1] For a further discussion, see M. Ross, "Machine Protection Schemes for the SLC," 1991 Particle Accelerator Conference.

[2] S. Clark, A. Grillo, N. Spencer, D. Hutchinson, J. Olsen, D. Millsom, G. White, T. Gromme, S. Allison, K. Underwood, M. Zelazny, and H. Kang, "Smart Machine Protection System," *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems.*

[3] M. C. Ross, "Beam Diagnostics and Control for SLC," 1987 Particle Accelerator Conference.

[4] J. T. Seeman, "The Stanford Linear Collider," SLAC–PUB–5607.

[5] M. C. Ross, N. Phinney, G. Quickfall, H. Shoaee, and J. C. Sheppard, "Automated Emittance Measurements in the SLC," 1987 Particle Accelerator Conference.

[6] M. C. Ross, J. T. Seeman, E. Bong, L. Hendrickson, D. McCormick, and L. Sanchez-Chopitea, "Wire Scanners for Beam Size and Emittance Measurements at the SLC," 1991 Particle Accelerator Conference.

[7] L. Hendrickson, S. Allison, T. Gromme, T. Himel, K. Krauter, D. Nelson, F. Rouse, R. Sass, and H. Shoaee, "Generalized Fast Feedback System in the SLC," *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems.*

[8] N. Heinen, N. Spencer, and J. Tinsman, "Improving the Reliability in the SLC Control System," 1989 ICALEPCS Proceedings.

[9] L. Hendrickson, N. Phinney, and L. Sanchez-Chopitea, "The SLC Correlation Plot Facility," *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems.*

[10] For example, M. C. Ross, N. Phinney, G. Quickfall, H. Shoaee, and J. C. Sheppard, "Automated Emittance Measurements in the SLC," 1987 Particle Accelerator Conference.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S01SRA05

Process Control for The Vivitron : the generator test set-up

J.R.Lutz, J.C.Marsaudon, R.Baumann, E.Kapps, R.Knaebel, J.Persigny.
Centre de Recherches Nucléaires, IN2P3-CNRS / Université Louis Pasteur
BP 20, F-67037 Strasbourg Cedex, France

Abstract

The VIVITRON is a 35 MV Van de Graaff tandem electrostatic accelerator under construction at the CRN since 1985. About half of the parameters are controlled by equipments which are highly stressed by their physical environment : sparks, electrostatic field, X-rays, vacuum, and gas pressure. It needs a dedicated process control system. The described control system is used since early 1991 to perform the voltage tests of the generator. It provides important information for the accelerator tuning and for the full size control under development.

I. THE VIVITRON

The Vivitron Van de Graaff tandem accelerator, under construction at the Centre de Recherches Nucléaires at Strasbourg France [1] is designed to reach a terminal voltage of 35 MV at its terminal electrode [2]. The tank (51 m long and 8.4 m in diameter) is filled with SF_6 at 8 bars. The charging system is a belt running close to the tube at a speed of 10 m/s. The column consists of a glass fibre / epoxy insulating assembly, supported by insulated epoxy posts. Seven porticos, large field-shaping shields, and discrete electrodes improve the electrical field homogeneity [6].

The expected energy will vary from 20 MeV/A for the light ions to 5 MeV/A for the heavy ions. The intensity should go from 10^{12} pps for the light ions to 10^9 pps for heavy ones.

II. THE FULL SIZE PROCESS CONTROL

A. Specific problems

The process parameters are spread over a large area. The control equipment, located at high voltage inside the accelerator tank and in the injector, are highly stressed by their physical environment : 35 MV breakdown flashes, 440 kJ stored energy, 1.7 to 10 MV/m electrostatic field, X-rays, vacuum, and SF6 gas pressure [3].

B. Architecture

A multi-level structure is implemented between the process and the operator.

Level 3 is in charge of the field equipment I/O interfaces, the handling, switching, buffering and communication of the I/O data. Some of these field equipment crates are located inside the accelerating tank and in the injector. They are connected to level 2 by optical-fibre links crossing the 2 MV/m electrical field. At least one crate is requested for each electrical equipotential level. The small space available in some “dead sections” imposes the choice of a small-scale bus crate.

Level 2 is located outside the vessel at ground level. It achieves communication with level 3 and with level 1 and provides data switching, concentration and handling.

Level 1 includes the communication interface, the real time control and the operator interface.

III. THE SET-UP FOR THE GENERATOR TESTS

For the generator tests, only a reduced process control is needed. No beam control is necessary, no automation is required, the number of parameters is limited and no equipment crate is needed inside the machine. All the information is fed out at both ends of the tank. Optical wires are used for sensors and activators at high voltage. Shielded and protected galvanic wires are used for those at the ground level. Thus, data acquisition, data switching and operator interface are dominant.

A. Current flow and terminal voltage

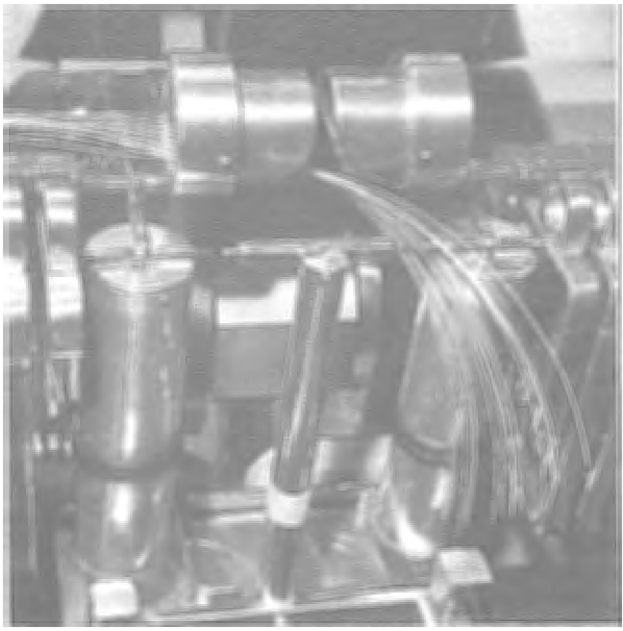


Figure 1. The current monitors in a dead section.

The most important features for the generator tests are the control of the terminal voltage and the current flow inside the accelerator tank. The GVMs (Generating Volt Meter), which register the electric field along the tank in a standard electrostatic accelerator, are inefficient because the internal electrode structure shields the field in the Vivitron.

Therefore, no beam being available, the terminal voltage is given by the sum of all the inter-electrode voltages. The only way to determine them is to measure the current flow along the calibrated resistor chains in each section. We developed therefore a powerless floating current monitor for the Vivitron (Figure 1), starting from an original Munich design [4]. The information is fed to the outside ground level by a plastic optical fibre. The measurement range extends from 1 µA to 1 mA with an accuracy of about ±1%.

B. Set-up

This reduced control system has been designed on the full scale control scheme but with some limitations (Figure 2).

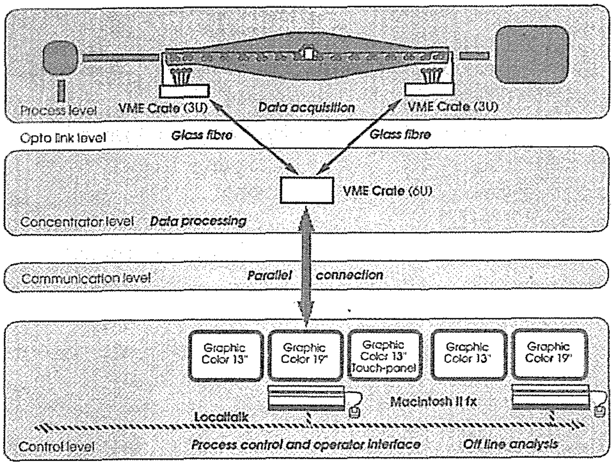


Figure 2. The control set-up for the generator tests.

Data acquisition and control is achieved by real time crates on levels 2 and 3, model of the final ones. Two diskless downloaded 3U VME crates on level 3, one at each end of the machine, are equipped with opto-isolated TTL I/O, 12 bits analog I/O, and timer/sequencer I/O boards. They communicate with level 2 by two serial glass fibre links. A 6U VME crate with hard disk on level 2 is in charge of the downloading of the former level 3 crates, of the message switching and of the data managing. The VME crates run under the OS9 real time operating system.

One standard, cheap, off-the-shelf available Macintosh IIfx 4/40 computer provides process control on level 1. It also provides the operator interface with multi-screen displays on one main 19' high resolution color screen (Figure 3), one 13' touch sensitive color screen (Figure 4) and one standard 13' color screen (Figure 5). A second identical computer near the first one is dedicated to the off-line data analysis (Figure 6) and the back-up. A third identical computer is devoted to the software development and to the process data base management. They are linked together by the Localtalk/Appletalk LAN. A Fastpath bridge provides access to a SUN server via Ethernet. They all run under MacOS. Communication between level 1 and level 2 is made by a NuBUS - VME Micron fast parallel interface. The two operating systems share a VME mailbox.

C. Tests of the generator

The tests of the generator started in the early 1991. A terminal voltage of 17.5 MV, the half of its nominal value, has been reached within a few weeks. Some tens of sparks occurred without disturbance of the process control.

About 100 parameters are controlled. Unidirectional and bidirectional currents are measured by about 60 current monitors and displayed as bar graphs on the central main permanent color screen (Figure 3). The column currents are displayed in real time as well as their differences in order to monitor the currents distribution and to locate current leaks. The belt charging currents and balance are also displayed graphically in real time. Intershield and terminal voltages are determined and displayed numerically on the same screen [7].

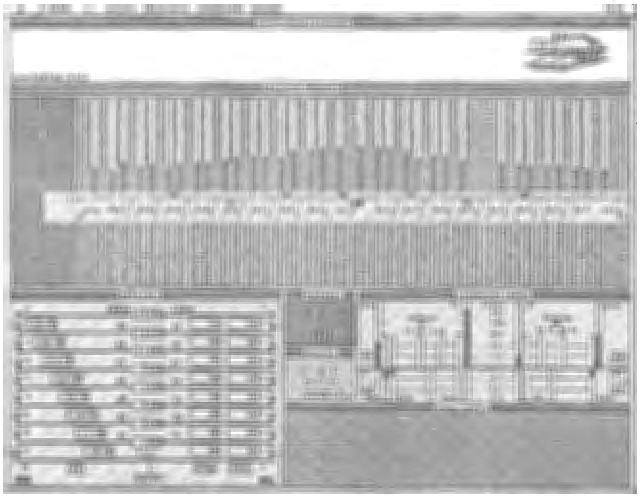


Figure 3. The operator main control display.

The control is achieved by the right hand touch sensitive color screen (Figure 4). Virtual push buttons switch the drive motors and the high voltage power supplies ON and OFF. They also control sliders which drive the charging systems. Interlocks are provided for security. The status is displayed by virtual green, yellow or red lights.

The left hand color screen is devoted to all the other functions : schematic display, wiring display, off-line and on-line

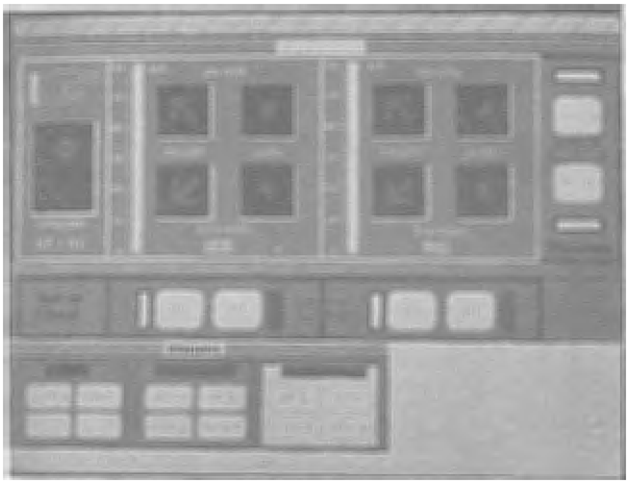


Figure 4. The command screen.

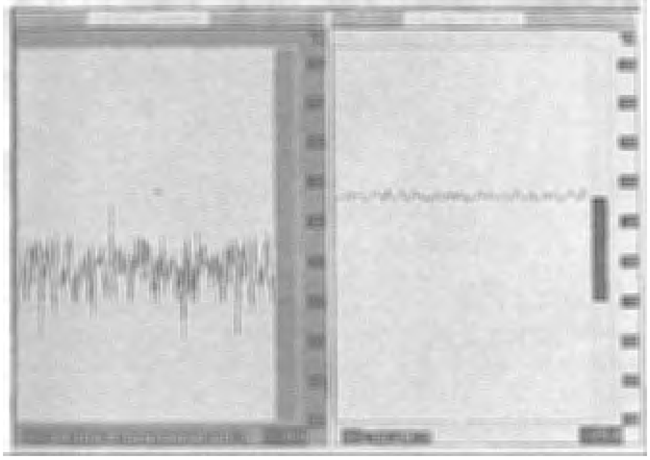


Figure 5. The real time current graph.

numerical data display, time chart of any selected parameters (Figure 5).

The refresh rate, of about 500 ms, depends on the computer load. All the data measured for 100 seconds are stored in a file with date and time on an operator command or automatically before and after a flash for later use and analysis (Figure 6). Every hour also, all the parameters are automatically stored on a backlog file.

The off-line analysis of the machine behavior and the study of the sparks can be achieved in two ways. The playback of on-line recorded data files by the control program performs continuous real time replay or step by step reading. The second way makes use of specific data analyzing programs or standard spreadsheet and graphic representation of data like EXCEL or WINGZ. A MACRO Language or Hyperscript are useful in this case. Figure 6 represents the currents flow inside the volume of the tank during a spark.

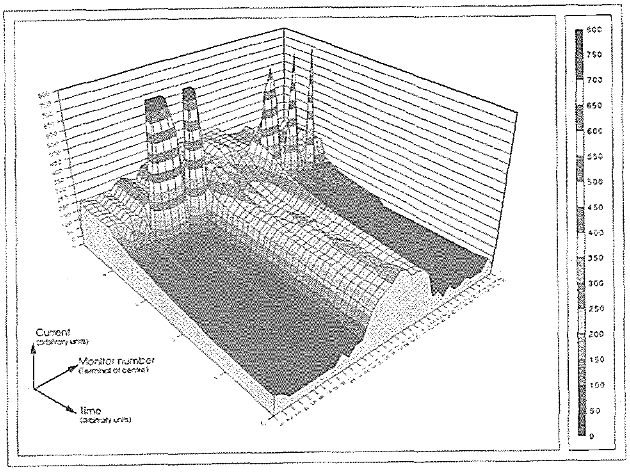


Figure 6. The 3D time / position current display.

IV. THE FUTURE

Equipment crates on level 3 will migrate inside the accelerator tank. The possibility to fit equipment crates inside the tank and their reliability have been tested in the MP accelerator. After more than one year and hundreds of flashes of up to 17 MV, we registered no transmission failure, no optical fibre failure, no measurement disturbance during flashes and only a few electronic failures [5]. The number of these crates will increase to more than ten.

The concentrator crates on level 2 will be connected to a high speed LAN and their number will grow. The process control and operator interface will be achieved by standard workstation clusters connected to the LAN. Processing power will grow from level 1 to levels 2 and 3. Software has to be more flexible.

V. CONCLUSION

The process control of the Vivitron had to be done in two steps. The first one has been used for the generator tests since early 1991. It represents a fast available control but it is reduced and limited in performance. Nevertheless, good experience is gained for the second step which concerns the full size control. Our aim is to perform a smooth transfer from the present status to the final one whereas the generator tests are going on.

VI. REFERENCES

[1] F.Haas, "Applications de techniques nouvelles à la construction des machines électrostatiques. Un exemple : le Vivitron", *Rev. Phys. Ap.* 23 (1988)1431
[2] M. Letournel and the Vivitron group, "The Strasbourg project. A 35 MV VIVITRON Tandem." *I E E E Catalog* n° 87CH2387-9, Vol.1, p.346
[3] J.R.Lutz, J.C.Marsaudon, R.Baumann, G.Frick, R.Knaebel,

C.Muller, J.A.Persigny, G.Rech, D.Stadelmann, "Contrôle et Commande du Vivitron ", *Rapport d'Activité 1987 Centre de Recherches Nucléaires*, F- Strasbourg

[4] L.Rohrer and H.Schnitter, "A light link coupled current monitor", *Rev. Phys. Ap.* 12 (1977) 1415

[5] J.R.Lutz, J.C.Marsaudon, R.Baumann, G.Biechel, E.Kapps, R.Knaebel, C.Muller, J.A.Persigny, G.Rech, D.Stadelmann, "MP tests for the Vivitron process control", *5th International Conference on Electrostatic Accelerators and associated Boosters*, 24-30 May 1989, F-Strasbourg - D-Heidelberg

[6] J.R.Lutz, J.C.Marsaudon, "The Vivitron process control", *International Conference on Accelerators and Large Experimental Physics Control Systems*, October 30 - November 3, 1989, Vancouver, B.C, Canada

[7] J.R.Lutz, J.C.Marsaudon, E.Kapps, J.A.Persigny, "The Vivitron process control", *2nd European Particle Accelerator Conference (EPAC)*, June 12-16 , 1990, Nice, France

High accuracy ADC and DAC systems for accelerator control applications.

E.Kuper, A.Ledenev
Institute for Nuclear Physics,
Novosibirsk, 630090, USSR

Abstract

In the work presented here the ways of construction, the apparatus for the precision measurements and control systems incorporated in the accelerating facilities of INP are considered. All the apparatus are developed and manufactured in the standard of CAMAC.

Introduction

While carrying the experiments on the precision measurements of the mezon masses on the installations with the electron - positron colliding beams one has to use the apparatus of a class 0.001% with the resolution about 0.0001%. An instability of the main power supply sources of magnetic systems of storage rings should not exceed 0.002%.

The powerful RF generators, the controlled sources of power supply with an output power of a few hundred kilowatts, pulse components of electron-optical channels, numerous digital devices including computers are the sources of different kinds of noises. Under these conditions, the stronger requirements on the noise damping are posed to the measuring and control equipment and to the analog data transfer lines.

In the power supply system of the facility VEPP-4 one has to measure of about two thousand points and to form the control signals for more than 500 channels. The time of energy rise is of a few tens of seconds. In the mode of operation one needs the high accuracy matching (0.1% - 0.01%) in the field variation in the magnetic components of accelerators. The technical parameters of the control and measuring structures should provide the operation of the power supply systems both in the static and in dynamic modes of operation.

Digital - to - analog converters

Usually, the power supply sources of the storage ring facilities requires the digital-to-analog converters (DAC) of quite a low fast action at an accuracy ranging from 0.1% up to 0.001%. Therefore, most of the converters used are designed on the base of the pulse width modulation PWM. The advantages of the given type of DAC are well known: the minimum of precise components at a practically arbitrary resolution, high linearity, easily achieved the galvanic isolation of the analog part and consequently, a low price.

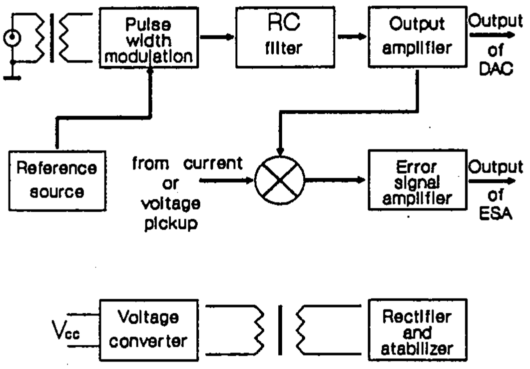


Fig.1. The analog section of PWM-DAC.

One of the popular developments of INP is an 8-channel code-to-duty factor converter (CDFC) located in the crate and transferring the control signal to DAC - PWM integrated directly to the control objects. The DAC signals in the form of different polarity pulse, the distance between carrying the data on reference voltage for control system are transferred through the coaxial cable with the transformer de-coupling to the distance of up to 500 m. The simplified schematic diagram of the converter is given in Fig.1. The pulsed signals from DAC arrive at the trigger controlled by the analog switches. The PWM modulated signal is filtered with the RC filter of the 3rd order. In order to match it with the control system the error signal amplifier (ESA) is envisaged which equalizer the DAC output signal to that from the current or voltage sensor. The galvanic de-coupling on power supply is performed with the help of high frequency converter with the transformer of special design with the minimum crossing capacitance. DAC parameters are: 16 bites, error - 0.01%, settling time - 0.4 s, temperature factor of the output voltage - 0.0003%/K. The given configuration is being widely used in the systems of pulse power supply of the transport channels for charged particles, in the power supply sources for the "high current" correctors, i.e. in those cases, where the controlled objects are distanced considerably and their groundings are explicitly non-equipotential.

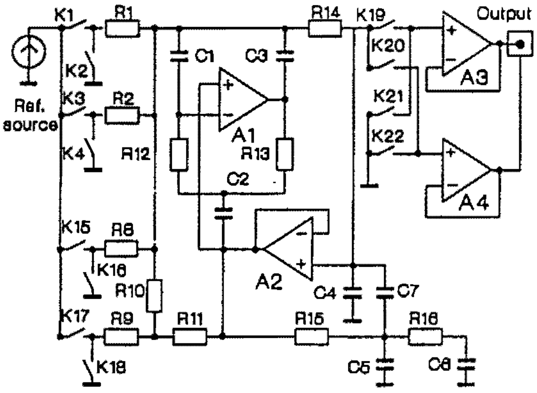


Fig.2. The analog section of multiphase PWM-DAC.

For the problems with higher requirements to the control accuracy (power supply systems of the main magnets and lenses of storage rings) the precision DAC was developed that is based on the use of the multiphase pulse-width modulation. This method is the improved version of the PWM. Its use enables one to reduce the settling time by a factor of the number of phases with respect to the conventional case and similarly to reduce the switching frequency. In this case, the requirements to the fast switching time of the analog switches determining the PWM signal became weaker. This fact enables one to simplify substantially the technical-design solutions of the switches and drivers using the standard logic elements of CMOS kind. The schematic diagram of the analog part of the apparatus is given in Fig.2. DAC is performed according to the two-cascade circuit. The output voltage is the sum (with weights 1/2048) of voltages of two independent DACs. The first one converts 11 senior bits and has an 8-phase generation of the output signal (switches K1 - K16). The second DAC for junior bits is a single-phase, 8-bit (K17 - K18). The voltages of both the DACs are summed by the resistors and smoothed by one filter of low frequency (A1, A2). The bipolar voltage is produced by the output circuit (K19 -K22, A3, A4). In the source of output voltage the precision reference diod is employed in the oven with the stability of temperature 1K.

The apparatus has the following parameters:

scale length	20 bits
voltage range	8,192 V
quantization step	15,625 μV
error (for 3 months)	0.001%
nonlinearity	0.0001%
temperature factor of output voltage	0.00002%/K +2μV/K
settling time with error 0.001%	0.1 s
analog part capacity with respect to the body	150 pF
module width	1 M

In the process of energy retuning of the storage ring of charge particles the matched variation of parameters is required for many power supply sources of magnetic elements with its high accuracy and highly synchronous. In the power supply systems of first generation this problem was solved by the appropriate selection (taking into account the individual characteristics of magnets) of special RC-filters on the DAC output. At present, the most relevant solution of the problem given is the use of DAC with the built-in digital interpolators. Two types of converters have been developed.

The multichannel DAC provides the conversion of a 16 bits code with the error 0.01% over 16 channels. The built-in processor makes the simultaneous variation of the output voltages. The variation law for each channel is given with the intermediate values by which the portion-linear interpolation is performed. Up to 80 intermediate values can be given for each channel. In addition, in each linear part the interpolation time is given within the range from 1 to 63 s with the quantization step of 1 s or in the range from 0.1 to 6.3 s with the quantization step of 0.1 s.

For the control of the precision channel a 20 bit DAC was developed with the built-in signal error amplifier. The conversion error is 0.001%. The functions of the digital interpolator are similar to that described above.

The converter control is performed through the controller with the protocol MIL-STD 1553B. The controller is designed in the CAMAC standard.

Apparatus for measurements of direct voltages

As already mentioned, the real operation of any large physical facility is usually accompanied by the generation of a large wide range of very different electromagnetic noises and quite often the amplitude of these noises is substantially higher than the signal to be measured. Therefore, in the analog-to-digital converters the most noise-resistant method of preliminary integration of the output signal is used.

Using various modifications of the method: the dual-slop integration, multi-tact integration, the method of dynamic integration, the series of ADCs is designed at INP aimed at the use in the multichannel noise-resistance measuring systems. Let us consider briefly the features of construction and parameters of the most characteristic versions ofthis series.

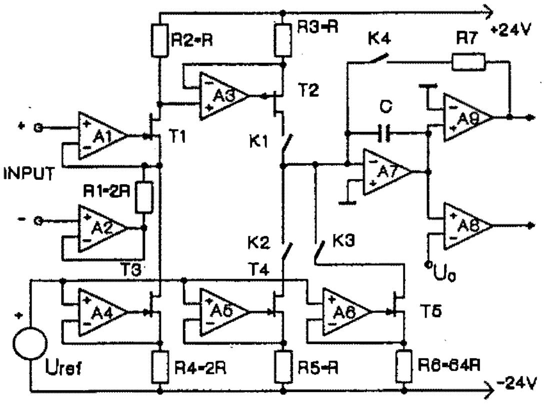


Fig.3. Three-step integrating ADC.

The three-step integration with respect to the dual-slop one enables the reduction of the error that is due to the noises of the integrator amplifier and comparator. An additional advantage of the method is its high resolution at comparatively low frequency of the tact generator. The simplified block diagram of the device is given in Fig.3. The input signal with the help of amplifiers A1 and A2 is converted into a current. A3, T2 is the current mirror. To achieve the fast action the switches K1 - K3 are made with diodes. The main reference signal is generated by the current generator on A5, T4. The reference signal of the third step 64 times lowered is formed with A6, T5. During the first step the integration of the input signal is performed. During the subsequent two steps the integration of the main and divided reference signals is performed. This method is realized on the ADC 15 - 256. Its main parameters are given in Tab.1. The device has a built-in memory for 256 words and the control circuit for the analog multiplexer that transfers the address of the measured channel in the subsequent code through the socket on the front panel. While performing the multichannel measurements the operation is performed in the following way: preliminary to the service register of ADC the initial and final addresses of channels are written along which the measurements should be performed. By the start command ADC performes the given series of measurements and writes the results into the corresponding cell of the built-in storage. The presence of this given mode enables one to reduce substantially the load of the CAMAC data bus in the measurement system.

The method of multistep integration enables one to reach the high resolution and linearity of the converter at quite high fast action. The block diagram of the method performance is given in Fig.4. Its essence is the following. Simultaneously with the input signal integration the reference signal is integrated by a certain algorithm. When the integrator output voltage reaches its threshold value (either A2 or A4 comparator is operated) to the integrator input the reference voltage is applied whose polarity is opposite to the input voltage for the fixed period of time, i.e. the multi-step integration of the reference and input signals is performed. After input signal integration the operation algorithm is the same as that in the previous method. The end of the 3rd step, during which the reduced reference voltage is integrated, is defined by the comparator A3. In this scheme the integrator transfer factor is approximately the same as the number of integration steps when measuring the input maximum signal.

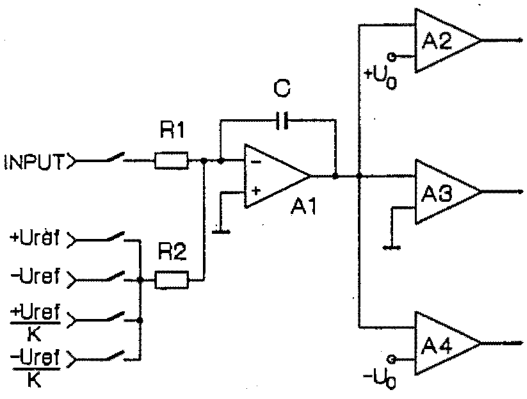


Fig.4. ADC with multicycle integration.

Note that conversion errors related to the polarization of integrator capacitor dielectric are reduced similarly. On the base of this method the precision converter ADC-22 was designed (Table 1). Let us give some additional characteristics of the device that are important for the construction of measuring systems of high accuracy: integration time of the input signal - 20 ms; settling time for the input amplifier (with an error 0.001% - 8 ms; signal measurement range 0.1; 1; 10; 100; 1000 V; resolution capability, respectively, 0.1; 1; 10; 100; 1000 μV; relative error of conversion in the range of 10 V for 8 hours - 0.0005% of the scale, an additional temperature error - 0.00005%/K; temperature drift of voltage - 0.03 μV/K; input current - lower than 10 pA; the input resistance on lower ranges - higher than 100 GOhm.

The method of dynamic integrator. This method is the version of the pulse width modulation PWM conversion with the pulse feed back. The block diagram of the dynamic integrator operation is given in Fig.5. The input signal is applied to the input of integrator A1 through resistor R2. Depending on the integrator input voltage polarity, defined by the comparator A2, an appropriate reference voltage is applied to the integrator input through the switches K1 and K2. (The operation of these switches is synchronized with the switching frequency by the D-trigger.) The specific feature of the method is that simultaneously with the input and reference signals through the capacitor C2 to the integrator input the periodic voltage is applied of the rectangular shape whose amplitude exceeds the sum of modules of the input and reference voltages. The main advantage of the method is that it enables one to vary the time and bits of conversion. This property is especially useful for ADC designed for the multichannel measurements: measurements with not very high accuracy are performed quickly but those of high accuracy - slower but with larger number of bits. With larger time of measurement an additional filtering is envisaged for high frequency noises in the measured signals.

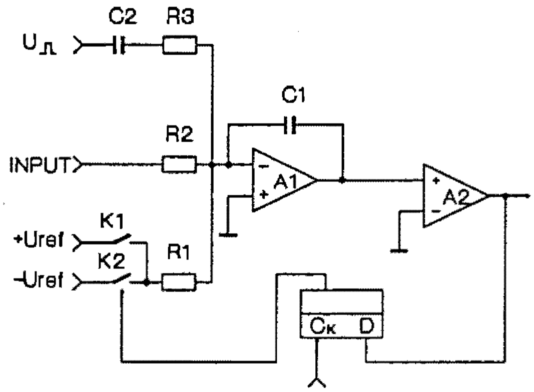


Fig.5. ADC with dynamic-integrator.

On the base of the method of dynamic integrator ADC-20 and ADC-20-256 (Table 1) are designed. Each of these modules has 8 time ranges of measurements (when switching the ranges the scale length changes respectively) and 2 ranges for the input signals (8 V and 500 mV) which enables the measurement of voltage in the microvolt range. ADC-20-256 has a built-in memory and the control for the multiplexer.

Table 1

Type of device	ADC22	ADC20	ADC20-256	ADC15-256
Conversion time, ms	40	7.5-480	1.25-160	0.1
Scale (binary)	22	14-20	13-20	15
Error (for 3 months)	0.001%	0.01%	0.01%	0.01%
Memory (words)	-	-	256	256
Common mode rejection, db	140	120	120	80

The metrological characteristics of devices remain the same within the range 20 K - 50 K.

For the arrangements of the multichannel measuring systems some analog multiplexers been developed at the Institute:

- AM-16R and AM-128R: with 16 and 64 channels respectively, the commutation elements - sealed contact reed relay with switching time of 1 ms, maximum voltage - 200 V, commutation error - 50 μV.
- AM-16RM and AM-32R: with 16 and 32 channels respectively, the commutation elements - thermocompensated reed-relays, commutation error - 1 μV. They are designed for the measuring systems in the microvolt range.
- AM-128: 64 channels, produced based on microcircuits with the complementary MOS-transistors, switching time - 10 μs, input voltage range - 10 V, commutation error - 100 μV.

The multiplexing of input signals is usually performed according to the two wire circuit.

In all the versions the protection is envisaged against the overvoltages by the input. The address register of multiplexers have 8 bits, that enables the union of up to 256 measuring channels per one converter.

The GSI Control System

U. Krause, V. Schaa, R. Steiner
Gesellschaft für Schwerionenforschung mbH, D 6100 Darmstadt, Germany

Abstract

The GSI accelerator facility consists of an old linac and two modern machines, a synchrotron and a storage ring. It is operated from one control room. Only three operators at a time have to keep it running with only little assistance from machine specialists in daytime. So the control tools must provide a high degree of abstraction and modeling to relieve the operators from details on the device level. The program structures to achieve this are described in this paper. A coarse overview of the control architecture is given.

I. THE GSI ACCELERATOR FACILITY

At GSI the heavy ion linac `UNILAC` runs successfully for 16 years. It produces beams of all elements up to uranium with energies between 1.4 and 20 MeV/u. In '89 the heavy ion synchrotron `SIS` and the experimental storage ring `ESR` started operation with ion energies up to 2 GeV/u.

A new control system has been designed for `SIS` and `ESR` which is adopted step by step also for the `UNILAC`.

The system has to handle quite different facilities,

— the `UNILAC` with 3 injectors and repetition rates of 25...100 Hz,
— the `SIS` with repetition rates of 0.1...3 Hz,
— the `ESR` with repetition rates of $\leq$0.001...0.1 Hz,
— transfer lines,

but has to provide an uniform operator access to all accelerators from one control room.

When the upgrading of the old components will be completed a total of 3000 devices scattered over an area of 250 × 250 meters have to be controlled. Complexity varies from simple DC magnets to ramped magnets (ramps generated from up to 900 samples) and diagnostic devices (up to 40kB of data). Fast switching of the machines allow time sharing between several experiments and injection to the next accelerator in sequence, all with independent beams.

II. OPERATING REQUIREMENTS

There is a need for several identical consoles for independent operation of accelerator segments and hardware redundancy.

A consistent "look and feel" has to be provided for the operator interface of all application programs.

To keep the control system serviceable and extensible it has to be a modular and uniform system with definite allocation of tasks. The size of the facilities suggests a decentralized, distributed and hierarchical system. Therefore real-time aspects are limited to the device handling level, data for these operations are stored there. By this means a clear separation between accelerator and device oriented aspects is achieved.

The operating level ("physics of accelerators"), located on powerful workstations, deals with the equipment as a whole and handles devices in a modeled form only. The device level ("physics of devices"), located mainly on microcomputers close to the devices, maintains and controls single devices. Both levels are connected by standardized device accesses using identical mechanisms for all devices.

III. CONTROLS COMPONENTS

On the operating level `VAX 3100-VMS` graphic workstations are installed forming a `VAX` cluster. Uniform software all over the system is ensured by cluster-disks. One to three `VAX`es are grouped as consoles for operator interaction.

The device level is equipped with two types of 68020 `VME` boards with a real time kernel. One is used as Master Processor (`MP`) for command evaluation. The other is equipped with an interchangeable communication interface and is used as Equipment Controller (`EC`) for device control and as Communication Processor (`CP`) for networking.

For communication between operating and device nodes `ETHERNET` is used. Devices are connected via modified `MIL-STD-1553B` serial bus with standardized InterFace Boards (`IFB`) as part of the devices. Besides this other interfacing like `IEEE-488` is partly used by simply replacing the communication part of the `EC` boards. Typically 5 to 10 devices are driven by one `EC`.

Time signals for process synchronization are broadcasted from a central timing system to all `EC`s via serial bus (`MIL-STD-1553B`) and Timing InterFaces (`TIF`).

Each `VME` node is equipped with `CP`, `MP`, `TIF` and 1...9 `EC`s. A diagram of the components is shown in fig. 1.

Actually 33 operating computers are used in the main control room and in several local consoles close to experi-

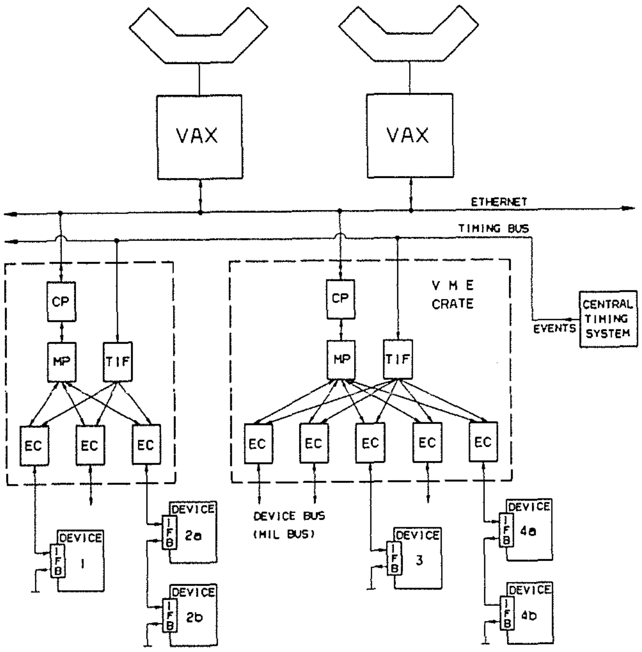


Figure 1: Hardware structure of the control system

mental facilities. For device controlling about 40 VME nodes with about 150 ECs are installed, driving more than 1200 devices.

IV. OPERATOR INTERFACE

For the variety of possible interactions with the control system, handling machines with a quite diverse nature, an operator console and operator interface has been defined combining every aspect of the operating modes.

The operator interface must offer a model of the process to be controlled. The process varies with the selected segment of the accelerator environment (SIS, ESR, UNILAC or transfer beamlines). Three software layers are present to support the selection mechanism to form the operating environment.

Layer 1 mainly consists of a code for selection of an accelerator segment. All computers forming a console are dedicated to the selected segment. Only codes which are usable in this context are provided, therefore all application, utility and service codes are grouped into categories. These categories are, e.g. usable at all consoles and accelerators, specific to an accelerator, to a beamline, a beamline segment or an experimental section. All specific codes serve only devices defined in the selected segment.

Additional codes are server for access to common databases for definition of application codes, accelerator and beamline segments, device names, properties, network addresses, error codes and help topics. Likewise alarm and error handling tasks and a logging facility for messages concerning starting and stopping tasks, abnormal conditions, failure of devices, are available in the console environment.

The second layer consists of two main process control codes, one for UNILAC and all beamlines, the other for ESR and SIS (SD[1], ZV).

— These application programs organize the access to devices using a number of specialized tasks. Each individual task uses the same context (i.e. same machine cycle, device setting etc.) and shares all process data.

— The acceptance of commands is acknowledged immediately, processed directly or passed to a subtask. All windows are organized by the central task and divided into areas with constant display of information and areas which are used by subtasks concurrently.

— The actual status of all devices in the selected segment is visible at a glance: device alarm (status or values changed) and appropriate refresh cycles guarantee an actual state of display. Icons reflect the current status of the device (switched on or off, positioned in beam or out, active or inactive etc.). Icons are unique for every device type. Additional windows are used for display of magnet read-outs showing deviations between actual and reference setting (see fig. 2), beam current profile and spill signals. History recordings offer an easy help to find a faulty element in case of beam–loss.

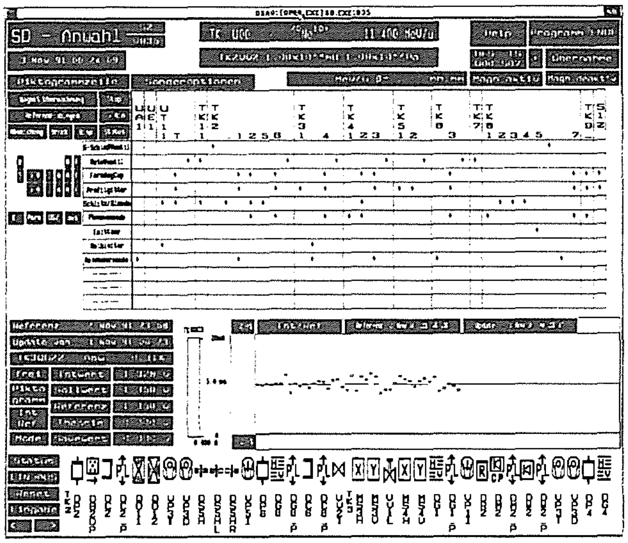


Figure 2: Process control for beamline

Additional codes of this layer are utility tasks, e.g. save magnet settings and restore saved or theoretical sets due to actual physical parameters, codes for manipulation of synchrotron devices according to algorithms, models and parameters, and for management and monitoring of the accelerator cycles. For supervision of the console environment (which task is running, on which node, network traffic etc.) a number of tasks is at hand.

The third layer consists of a number of specialized tasks which are managed by the main process control codes. Two groups are discernible, first codes used for supervisory control and update of device information. These tasks do not need output devices, servicing purely all other, sharing the same process data structures. The second group consists of interactive tasks like emittance measurement, isotope and charge spectra, beam profile display, etc. needing input/output devices as there are graphics, knobs, beam current control monitors.

All application programs provide a consistent "look and feel" by a standard color scheme to present device status and action. There is an unique screen layout management for programs on graphic screens and terminals. Selection fields performing similar actions are standardized in colors, position and names. An uniform access mechanism for all devices and a transparent handling of processes and functions guarantees that the operator does not need the knowledge about the actual control sequences.

Beside the operating environment, for troubleshooting, debugging, test and running-in `NODAL` [2] is widely used by maintenance people.

V. DEVICE MODELING

To ease handling of the big number of devices they are classified by types (DC magnet, pulsed magnet, beam position monitor, rf-structure, ...). All devices of the same type are represented identically to the operating level and use the same software on the device level. Different device parameters like maximum current of power supplies or number of wires of a position harp are considered by device specific constants in databases on device level.

The set of relevant features of one device type is called Equipment Model (`EM`) of which actually 42 are supported. They are modeled and described in a standardized way to allow the same access mechanism for all devices.

Every device is identified by an unique name of 1 to 8 alphanumerics, called nomenclature. Each feature of a device of every `EM` is represented by so-called property and property class. The properties are described by a name of 1 to 8 alphanumerics, the property class as designation of data exchange (`R` read from device, `W` write to device, `N` no data exchange), and a description of the data associated with the property: the number of data to exchange and their representation (`INTEGER`, `REAL`, `BITSET`, ...).

For a magnet the feature "set value of output current" would be described as

property: `CURRENTS` (`S` indicates set value)
property class: `W` (data send to device)
data count: 1
data type: `REAL`

Properties are `EM` dependent, but standard ones like mains switch (`POWER`), device status bit pattern (`STATUS`), reset (`RESET`), etc. are identical for all devices.

Accessing devices from the operating level means simply calling the associated property/property class with exchange of data for a device represented by a nomenclature. For greater flexibility several ways of calling are supported like execution with or without response (at least acknowledge) from the device level, synchronous (wait for response) or asynchronous (response may be read later), automatic command execution periodically, etc.

Before sending a command to the appropriate controllers the operating representation is changed to the device representation. This includes the transformation of the device nomenclature to addresses used for identification on the device level, property/property class to the identifier of the associated software module (see section VII.) and data from operating to device format since both may be different, i.e. `REAL` on operating level, `INTEGER` on device level.

Data in the responses to a command are converted from device to operating representation and then the response is supplied to the user or buffered for later reading.

Device accessing is handled by one universal software module, using the same mechanism for all devices. All `EM` and device informations needed are extracted from a Central Data Base (`CDB`).

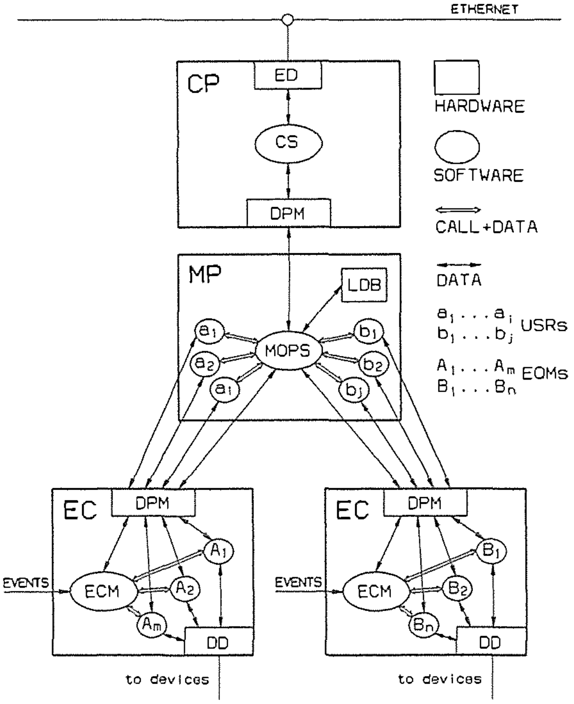


Figure 3: Software structure on the device level. `ED`: `ETHERNET` driver, `DD`: device bus driver, `DPM`: dual ported memory, `LDB`: local data base

VI. DEVICE CONTROLLER LEVEL

Device controlling in the closer sense of generating the command sequences for remote control is realized on the `EC` only. This is the only level where real time requirements have to be fulfilled. To enhance the power and to facilitate fast reaction with well defined response time the task of the `EC`s is limited to device driving only without interruption by higher levels. All data needed by the device are stored on the `EC` and it runs completely autonomously. Communication with higher levels is by dual ported memory on the `EC` and thus may be carried out asynchronously.

On the EC is coded *what* to do. Timing requirements (*when* to do) are managed by the central timing system which delivers 255 different time signals, so–called events, to all ECs.

Activities are coded in EM specific **EQ**uipment **M**odules (EQM) as the smallest independently executable units on ECs. They may contain complex actions consisting of several device commands. EQMs are connected to events with an identical connection for all devices of an EM. Execution of EQMs is on reception of the associated event. This guarantees systemwide synchronous operation of otherwise independent devices if only connected to the same event.

Besides event driven EQMs other ones may be called by direct request from higher levels signaled in the ECs dual ported memory. Execution is not immediate but at command events always delivered in the gaps between accelerating cycles when real time reaction is not required. For pure DC devices no events are evaluated, all actions are executed directly when initiated by operating request.

EQMs cannot be interrupted so execution on the EC is strictly serial. If an EQM could not be started in time since a previously started one is still running, an error is signaled allowing EM dependent handling.

Event reception and calling of EQMs is done by the **E**quipment **C**ontrol **M**onitor ECM, installed on every EC. Furthermore it supervises the status of the EC and the connected devices.

Process control by the timing system enables an easy way of fast switching between different accelerator cycles. To do so for every device 16 datasets for different accelerating cycles are stored on the EC. Selection and thus establishing the generated cycle is by a dataset number as part of the delivered event code. So the timing system determines the sequence of accelerating cycles with switching from pulse to pulse. This allows alternative supply of several experiments, each with different beams, including the injection to the next accelerator in sequence.

For a detailed description see reference [3].

VII. COMMAND LEVEL

The task of all components between the operating and the EC (implemented in hardware as well as in software) is to link the operational representation of devices to the device handling on an EC. ECs are linked to the operating by the MP to not burden the EC. The additional communication processor CP only manages the ETHERNET handling and is mainly transparent.

The MP has to evaluate commands from the operating level and to send back the responses.

EM specific command evaluation is coded in modules, called **U**ser **S**ervice **R**outines (USR). Every existing combination of property/property class/EM is represented by one USR on every MP where the EM is realized. Calling a property/property class simply results in execution of the associated USR. An USR may do any combination of data processing like evaluation of measurements, sending data to EC or receiving data from it, calling an EQM or another USR and performing consistency checks of exchanged data.

USRs are called by the **M**aster processor **OP**erating **S**ystem (MOPS). Since commands are sent in standardized form, analysis, dispatching and sending results to the operating level is the same for all devices and can be handled completely within MOPS. The handling of connected commands, i.e. periodically calling an USR, is implemented in MOPS, too. MOPS supervises and displays the status of MP and assigned ECs. Extensive multitasking allows quasi-parallel execution of commands on the MP level.

VIII. AUTOCONFIGURATION

Nomenclatures, device numbers and property description are stored on device level in local data bases, one per MP. From these databases the CDB, needed for device accessing, is automatically updated:

— At startup of an operating VAX it collects the local databases of all MPs to form the CDB,

— At startup of a MP the local database is sent to all control VAXes for update of the CDB.

A hierarchical still–alive supervision is implemented on every level of the controls system.

IX. CONCLUSION

The control system is in use since commissioning of the new facilities three years ago. During this period no severe problems were encountered. Routine operation of the machines now shows a reliable system. This experience confirms the structural concepts of the system being flexible enough to implement additional requirements and features.

Major effort now is to upgrade the old facilities and to install additional beam diagnostics both resulting in adding a lot of devices to the system. On the operating level main focus is in clarifying and unifying the complex handling and thus enabling a simpler and more reliable routine operation.

We would like to take the opportunity to thank all our colleagues who have contributed to the success of this control system.

X. REFERENCE

[1] V. Schaa, G. Fröhlich, M. Balloff, Jun–Ye Wang, "A New Generation of Operating Programs for the Accelerators", GSI Scientific Report 1987, p. 357

[2] M.C. Crowley–Milling and G. Shering, "The NODAL System for the SPS", CERN 78-07 (3 September 1978).

[3] R. Steiner, C. Riedel, W. Rösch, "Real Time Aspects of Puls to Puls Modulation", this conference.

VME Applications to the Daresbury SRS Control System

B.G.Martlew,M.McCarthy,W.R.Rawlinson
SERC Daresbury Laboratory,Warrington WA4 4AD UK

Abstract

The control system for the Daresbury SRS has recently been extended with a VME based alarm system which is operational. A further development is a steering system to provide servo control of the electron beam orbit position in the storage ring.

I. INTRODUCTION

The Daresbury SRS is a 2 Gev electron storage ring dedicated to providing synchrotron radiation to approximately 32 stations on 10 beamlines. It came into operation in mid 1981 and was upgraded with a high brightness lattice in 1987.

The control system for the SRS was designed and constructed in the period 1975-1980 and the original computers were upgraded in 1985.

The original control system provided an alarm condition monitoring system with a sampling resolution of 2 minutes. Recently, a dedicated VME system has been added which provides alarm monitoring and indication with sampling at the level of 5 seconds.

The high brightness lattice upgrade in 1987 reduced the source size by a factor of 10 and this led to difficulties with beam alignment and positional drift over the period of a stored beam. Work is under way to provide a VME based beam steering system to provide servo control of the electron beam position.

This paper will give a brief description of the SRS control system followed by a description of the new alarm system. A description of the beam steering system and its present status will be given.

II.THE SRS CONTROL SYSTEM

The SRS control system consists of a network of 4 Concurrent Computer Corporation(CCC) 3200 series computers as shown in Fig 1.

All computers in the network have a CAMAC system crate and communicate via CAMAC fast serial data links. CCC3230 is the operator interface computer providing service to three operator consoles in the main control room via a CAMAC parallel branch. Two of the operator consoles contain a colour display and keyboard (Tektronix 4207), a knob and tracker ball and a monochrome graphics display. The third console contains a Tektronix 4207 colour display and keyboard and serves as the personnel safety console.

CCC3205A and CCC3205B computers provide the interface to the plant via serial CAMAC highways. CCC3205A has two serial highways, one for the linear accelerator and Booster synchrotron and one for the Storage ring and Beamports. CCC3205B provides control for the beamlines with one serial CAMAC highway. Local control at the plant is implemented with Tandy 102 computers. The total parameter count presently stands at approximately 1800.

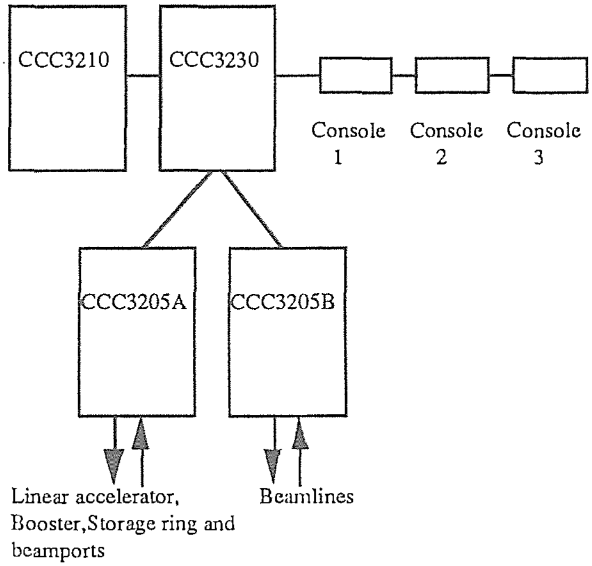


Fig 1. The SRS Control system

CCC3210 is offline to the control system and is used for programme development and testing.

High level programming is done in RTL/2 with more recent applications written in C.

III. THE ALARM SYSTEM

The original SRS alarm system was in operation for 9 years from the start up of the machine. Over this period, the system has been found to have several disadvantages:-

1. Noisy analogue input signals gave spurious alarm indication which led to alarms being ignored by operators.
2. Alarm conditions were applied to large numbers of parameters which led to 'swamping' of the alarm display and the operators being presented with more information than they could reasonably handle.
3. The large number of alarmed parameters led to difficulties in administration of the system.
4. The 2 minute sampling rate meant that the system was in fact no more than a fault indication system rather than a true alarm system.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S01SRA08

5. The monochrome,text-only display did not provide an eye-catching indication of alarms.

Whilst it would have been possible to improve the existing system for points 2. and 3., the system could not have been enhanced to address points 1. and 4. without seriously effecting control system performance. It was therefore decided to produce a new alarm system with separate intelligence. The new system was to have the following features:-

1. The number of alarmed parameters would be kept to a relatively small number of important parameters.
2. Faster sampling of alarmed parameters was required.
3. Operator interaction with an alarm display was essential.
4. Audible alarms in the Control room were desirable.
5. Alarm history for at least the last 24 hours should be available.
6. A hard copy of alarms was required.
7. A clear,unambiguous display of alarm conditions was essential.

A VME based system was chosen both for its long term expandability and to allow us to gain experience with VME and the chosen operating system,OS/9.

The hardware chosen consisted of a Motorola 68020 processor running at 16MHz with floating point co-processor, 4 Mb Ram, 5 RS232 channels, interface for high resolution coloured graphics device, keyboard and mouse and 40 Mb hard disc and floppy disc drive. The system is shown in Fig 2.

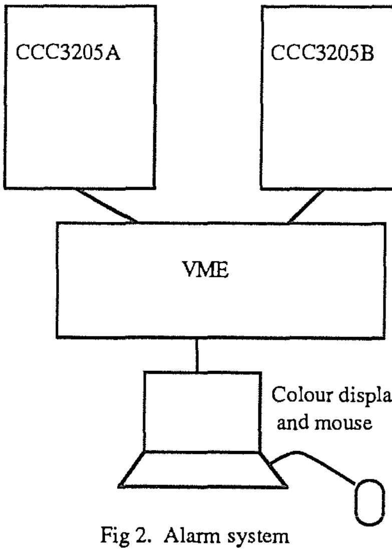


Fig 2. Alarm system

CCC3205A and B are connected to the VME crate with RS232 links operating at 9600 baud. A future improvement will be to remove the main control system CAMAC fast serial data links and replace with an Ethernet LAN to which the alarm system would be connected. Simple, low priority processes in the 3205 computers monitor a table of parameters every 5 seconds and transmit raw analogue and status data over the RS232 links to the VME crate.

A matrix type display consisting of various zones corresponding to different machine areas, forms the alarm system indication. Each box on the matrix is normally depicted in black and white with a change to flashing red when an alarm for that zone is detected. By selecting the flashing zone with the mouse, the operator is presented with detailed information on the alarms in that zone and flashing ceases although the box remains red until the alarm is cleared.

The software is arranged as a ring of processes communicating via OS/9 signals as shown in Fig 3. Data from the 3205s is transmitted over the RS232 links to the input processes and piped to the alarm monitor process which performs alarm checking and passses information for display and logging to the display manager and then onto the disc logger for history storage and to the print logger for hard copy. All processes have access to the alarm system database. The alarm monitor process checks supplied analogue and status data against limits in the database to determine the presence of an alarm condition. Warning and danger levels may be specified for the alarm condition. The operator may request the system to ignore or notice individual alarmed parameters. Future extensions to the system include the implementation of rate of change alarm conditions and the provision of audible alarms.

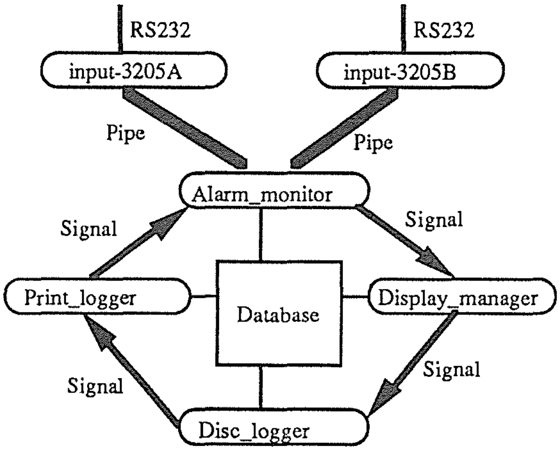


Fig 3. Alarm system software

IV. BEAM STEERING SYSTEM

Since the upgrade of the SRS to a high brightness configuration in 1987[1] there has been a growing requirement from the user community for improved beam position stability.

Beam stability measurements on the SRS and requirements for a new steering system have been described elsewhere[2]. Vertical beam position stability is most critical for users, with the largest movement being a peak displacement of 250±100µm in the first 3 hours of a beam fill followed by a roughly linear decay of approximately

20µm/hour for the remainder of the fill which typically lasts 24 hours. The existing electron beam monitors have an accuracy and repeatability of ±50µm and are capable of providing only gross orbit correction. A two wire photon beam monitor has been used at Daresbury to monitor photon beam position and has shown a useful range of 3mm with a resolution of 10µm at 21m from the source[2].

Steering elements on the SRS consist of 16 vertical steering magnets, 16 dipole trims produced by backleg windings on the main dipole magnets and 16 multipole magnets each made up of 12 individually powered windings which provide a combination of horizontal steering, vertical steering,octupole and quadrupole fields. In total there are 224 individual power supplies which are driven from 12 bit CAMAC DACs.

Planned improvements to the steering system are:-

- Upgrade of the steering magnet supply DACs from 12 to 16 bit resolution and accuracy.
- A VME based system to provide local intelligence and increased bandwidth for steering magnet servo control.
- Production of tungsten vane monitors for photon beam position monitoring.
- Provision of improved electron beam monitors having a resolution of 10µm[3].

The aim is to achieve a photon beam stability of 50µm at 20m from the source.

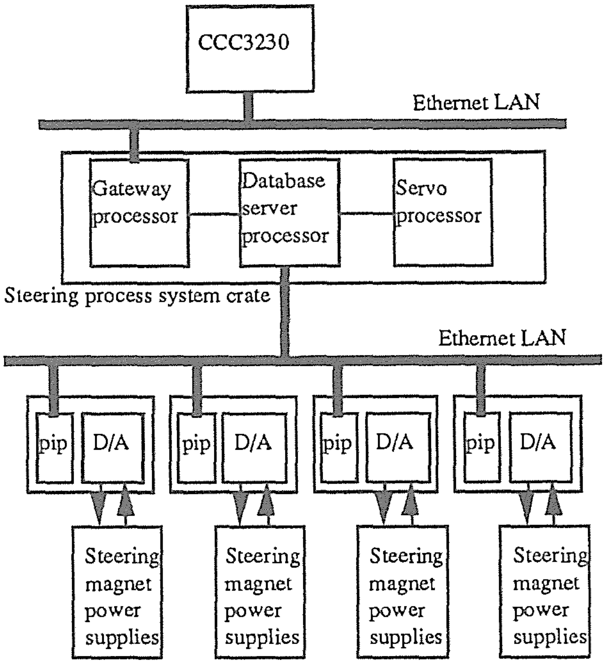
V. VME-BASED STEERING SYSTEM

The new steering system will involve disconnection of the steering magnet DACs and ADCs presently in CAMAC serial crates and connection to VME DACs and ADCs housed in VME plant interface crates. Fig 4. shows the arrangement of the new system.

The Steering Process system crate contains three processors, Gateway, Database server and Servo. The Gateway processor is the interface to the existing control system and contains code which makes it appear as another 3205 processor to the 3230. The 3230 and the Gateway processors are nodes on a dedicated Ethernet LAN. The Database server processor contains the steering system database and is connected to another Ethernet LAN which also has plant interface crates as nodes. The Servo processor contains global and local feedback algorithms. All three processors have access to the database which is held in shared memory. The plant interface crates contain a processor (PIP) and enough DACs and ADCs to service one quarter of the storage ring steering magnets. In addition, these crates will have ADC channels to read in signals from electron and photon beam monitors and various environmental parameters.

The Gateway processor has access to the hard and floppy discs while all other VME processors in the system are ROM based systems. The database will be continously updated by a process in the Database server communicating with the plant interface processors.

The processors are MVME147s with a Motorola 68030 cpu with a clock speed of 25MHz, 8 Mb RAM, 4 serial and 1 parallel ports and floating point processor. There is an 80Mb Winchester disc and 1Mb floppy drive in the steering system process crate accessible to the Gateway processor.



pip: Plant interface processor D/A: DACs and ADCs

Fig 4. VME based steering system

The Gateway processor runs under Professional OS/9(V2.4) while all others run under Industrial(ROM based) OS/9. Application software is written in C. The communication protocol is TCP/IP.

VI. PRESENT STATUS AND CONCLUSIONS

The new steering system hardware has been purchased apart from the DACs and ADCs which are presently being evaluated. The interface software both in the 3230 and Gateway processors has been written and tested and the Database structure has been established. Work is progressing on the processes in the Database server and Plant interface processors for database updating. Further work is necessary to design and write code for the Plant interface processors. It is intended to install and test a minimal system without the new electron beam monitors in 1992.

The power and versatility of VME make it a strong contender to form a major part of a future accelerator control system at Daresbury.

VII. REFERENCES

[1] V.P.Suller et al., "SRS-2: Performance and Achievements" presented to the Particle Accelerator Conference, Chicago, March 20-23 1989.

[2] P.D.Quinn and T.Ring, "Developments in orbit control at the SRS at Daresbury", presented to ABI Conference, KEK, Tsukuba, Japan, April 22-24 1991.
[3] T.Ring and R.J.Smith, "Orbit measurement techniques at Daresbury", presented to ABI Conference, KEK, Tsukuba, Japan, April 22-24 1991.

Accelerator Control Systems in China

Yao Chih-Yuan
Hefei National Synchrotron Radiation Laboratory
U.S.T.C.

Abstract

Three accelerator facilities were built in the past few years, the 2.8 GeV electron positron collider BEPC, the heavy ion SSC cyclotron accelerator HIRFL and the 800 MeV synchrotron radiation storage ring HESYRL. Aimed at different research areas, they represent a new generation of accelerator in China.

This report describes the design philosophy, the structure, performance as well as future improvements of the control systems of the these facilities.

I. INTRODUCTION

The development and research of accelerators in China has made good progress in the past thirty years. Many low energy accelerators for research and application have been constructed, including high voltage type accelerator, cyclotron, linear accelerator, betatron etc. Their application covers a wide range: medical treatment, industrial irradiation, non-destructive inspection, isotope production and many other fields. The three newly completed high and medium energy accelerator facilities, Beijing electron positron collider(BEPC), Lanzhou heavy ion research facility(HIRFL) and Hefei synchrotron radiation source(HESYRL), aimed at different scientific research areas, represent a new generation of accelerator facilities.

Early accelerators are mostly controlled by panel meter and push button type manual control system. Application of microcomputers to accelerator control system started at the beginning of 80's when microcomputers were becoming popular in China.

This paper is not intended to be a general survey. Control systems of the three new accelerator facilities, which are relatively larger in scale and more complicate in structure, are described here with the emphasis on the system architecture.

II. BEPC CONTROL SYSTEM

BEPC is the first high energy accelerator built in China. The main facilities of BEPC are a 1.4 GeV electron linear accelerator injector, a 1.4 GeV beam transport line and a 2.8 GeV electron storage ring. The project was started in 1984 and completed in 1989.

Because of the strict time table for the BEPC project, the leaders of the project decided early in 1985 that in order to reduce development work and shorten the construction period the new control system of SPEAR ring should be adopted as the base for BEPC.

BEPC control system is a typical centralized control system. A VAX11/750 serves as the central control computer. Serial CAMAC systems are the base for equipment interfacing.

1. System Configuration

Fig. 1 is a block diagram of BEPC control system. The system is functionally divided into three levels.

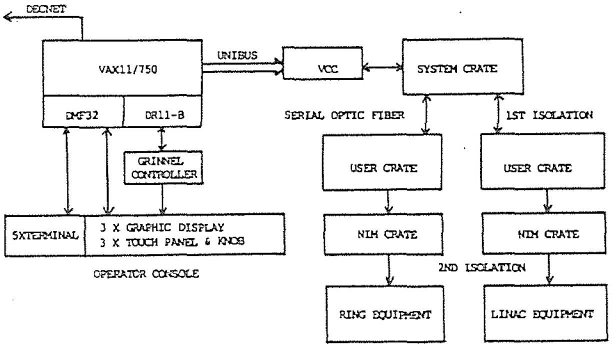


FIG.1 BLOCK DIAGRAM OF BEPC CONTROL SYSTEM

First, at the center of the system is the DEC VAX11/750 computer, which is equipped with a asynchronous serial interface board for connecting terminals and knobs, a DR11-B DMA interface for connecting color display monitors, and other standard peripherals such as hard disks, printers, tape drivers etc. A SLAC designed Vax CAMAC Channel(VCC) is the key element in data communication network. It is a DMA controller to interface VAX11/750 UNIBUS with CAMAC system. Two CAMAC system crates are controlled by the VCC, one for the linear accelerator the other for the storage ring. One system crate houses several serial branch driver modules and each branch driver starts a fiber optic serial high way loop which connects up to 7 user CAMAC crates. The second level is the local control stations formed by user crates. I/O functions are performed by the local control stations. Each local control substation controls one section of equipment, such as ring magnet system, vacuum system, transport line magnet system, RF system, linac magnet system, BPM system etc.

At the third level are NIM signal converter and isolator modules distributed at equipment site. The NIM modules must match BEPC hardware and are all developed by IHEP.

2. Software System

BEPC control software is a database driven system taking full advantage of multi-tasking ability of VAX/VMS operating system.

The database and its manager are the center of the system. All the machine hardware and program operation parameters are stored in the database which resides in a VAX global section shared by all software processes.

Software processes are executable programs under VMS operating system. They can be invoked by terminal command or by another process.

Two processes have special role in the system. A memory resident VCC I/O program CXCAMAC continuously performs I/O operation, sends out messages to CAMAC systems, collects status information from CAMAC systems and refreshes database records. The other special process is AVTX, which is a command receiver/interpreter and scheduler. It receives touchpanel and knob commands, decides which processes to call to perform the demanded operation, and activate appropriate processes.

Database generation and maintenance processes are also included in the system for ease of database initialization and modification.

Up to now, a total of 17 software processes are implemented for BEPC system, including ring magnet control and status display, ramping process control, BPM data taking and processing, closed orbit correction, ring lattice calculation, ring and transport line modeling etc.

3. Present Status

The system was completed in the middle of 1988, a few months ahead of first colliding beam experiment. After that it has been operated for beam colliding experiment and machine study for more than two years. System reliability is proved satisfactory. Some modifications has been made to the system to improve its performance. Dual speed Dual accuracy ramping function are developed for main magnets and correctors to increase the speed of ramping process. Analog control part of database is reorganized to enable more flexible and faster I/O operation. RF control function is added and a new BPM data acquisition program is developed to reduce BPM scan time and generate graphic display of beam position.

4. Upgrading Plan

Because of the centralized architecture, all control and monitoring operations must be initiated by VAX11/750. This takes substantial part of CPU time, especially during a ramping process. Many memory resident processes further increase the load of VAX11/750. The processing speed and response time of the entire system are not satisfactory. Further more, system reliability to some extent depends on the reliability of VAX 11/750, as it is the only controlling master. Another problem is that because of its limited processing power, VCC sometimes becomes the bottle-neck of message communication.

In order to overcome these problems, plan has been made to convert the centralized system to a distributed system. DECNET will be used to link the VAX11/750, A micro VAXII and a VAX Work-Station as three nodes. Fig.2 shows the upgraded system configuration.

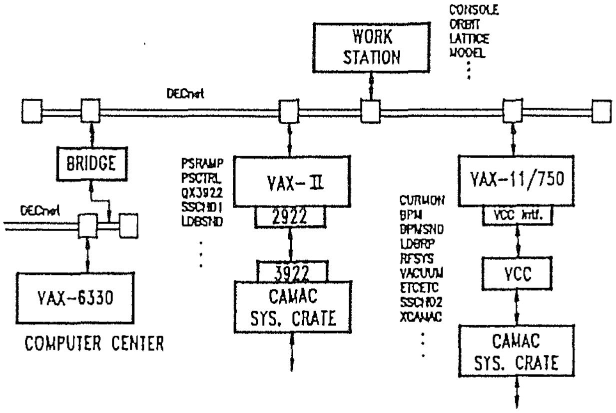


Figure 2 The upgrade of BEPC control system

The Micro VAXII will be dedicated to the task of magnet power supply control. The WS will be used to replace the present console terminals. The rich window software functions of WS will be fully utilized to update the man-machine interface. The WS will also share some of the calculation tasks, such as ring modeling, orbit calculation etc., with the VAX11/750.

A KSC 3922/2922 Q-BUS CAMAC adapter will be used to interface the Micro VAXII to the system crate for power supply control. The communication bottle-neck problem will be basically resolved, partly because the amount of messages through VCC are reduced substantially, partly because the QIO operation of 3922 is faster.

All hardware below the system crates will remain unchanged. This will ensure hardware compatibility.

The software compatibility among VAX family systems is an advantage for software upgrading. Except for the CAMAC I/O driver program, most of the software developed for VAX11/750 can be moved to Micro VAXII system and the WS with only minor modification.

In order to coordinate the processes on

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S01SRA09

different nodes, new communication programs will be developed to enable inter-node and inter-process message exchange.

III. HIRFL CONTROL SYSTEM

HIRFL is a variable energy heavy ion accelerator. It consists of a 1.7 meter radius sector focusing cyclotron modified from a former 1.5 meter cyclotron as injector and a new built separate sector cyclotron. The project was started in 1976, and completed in 1988.

Control system of HIRFL is a distributed control system formed by a Vax cluster as the central computer and several microcomputer controlled CAMAC control substations(CSS).

1. System Configuration

Fig. 3 is a block diagram of the HIRFL control system. Two VAX8350 computers are linked together through the CI bus and DECNET to form a cluster. One of them, which serves as the central computer of SSC control system, interfaces to serial CAMAC system through a UNIBUS adapter and a serial branch driver. The other serves as a data processor, software development system and a backup system. The cluster is equipped with 2X12MB of memory, 4X520MB hard disks and other standard peripherals.

A DZ11 asynchronous interface, a DR-11 parallel interface and a parallel CAMAC branch driver are also installed on the UNIBUS for connecting console devices, including 6 touch panel monitors and 6 color monitors.

Serial CAMAC high way communication link is used for the whole control system.

A control substation is an intelligent CAMAC crate containing a serial crate controller which is the main controller of the crate an auxiliary controller for substation computer and other CAMAC modules. Control command may either be issued by the central computer or by the substation microcomputers. Communication between the central computer and substation computers is also performed via the serial high way. A substation can be further expanded through a secondary serial CAMAC high way loop if more than one crates are required. Several types of control substations have been constructed during system development period. In order to reduce the number of different subsystems, only two of them, the PC based and LSI-11 based CAMAC control substations are adopted for present system.

Accelerator equipments are divided into several subsystems. They are controlled either by normal serial CAMAC crates or by control substations.

The injection and extraction transport line and SSC magnet power supplies are all controlled by serial CAMAC crates which are directly connected to the main high way loop and controlled by the central computer.

RF system is controlled by a PC based CAMAC control substation composed of 8 CAMAC crates.

Vacuum system is monitored by another PC based CAMAC substation.

Beam diagnose system is controlled by a LSI-11 based CAMAC control substation composed of 4 CAMAC crates.

All equipments are connected to CAMAC modules via signal conditioner electronics which was developed by HIRFL.

2. Software system

The software of HIRFL control system has two levels. The main control programs for the central computer are all VMS tasks and

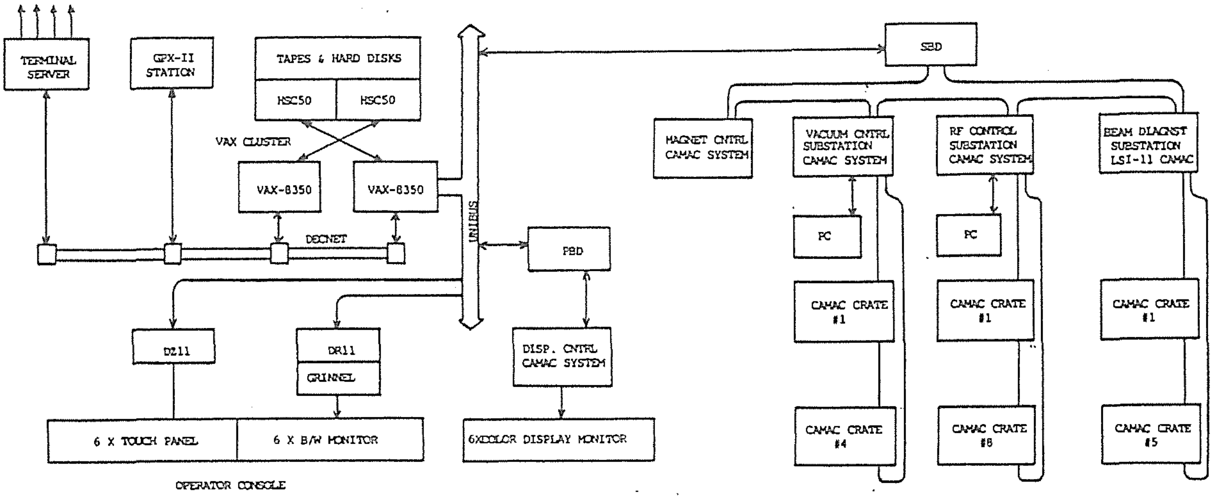


FIG.3 BLOCK DIAGRAM OF HIRFL CONTROL SYSTEM

written in FORTRAN. They accept main console command, generate various displays and send control command and data to CAMAC crates or to the control substations.

Software for the substations were written with FORTRAN and BASIC languages. Menu type man-machine interfaces are implemented for convenience of operation. The substation control programs perform all the basic control functions: equipment parameter adjustment, on/off control, status monitoring and generation of local status displays.

3. Present Status

All the control substations were completed in 1988 and was in operation before the beginning of SSC commissioning. They have satisfied the requirement of HIRFL commissioning and operation.

A system composed of two 386 PCs and CAMAC systems were built for the SFC control recently.

The VAX cluster was installed in Jan. 1988. A VAX control program for SSC magnet power supply system is completed and in operation. Development of VAX central control programs for other subsystem is under way.

The next goal of HIRFL control group is to realize central control of all SSC and the SFC equipments.

IV. HESYRL CONTROL SYSTEM

HESYRL is a dedicated synchrotron radiation source designed for UV and X-ray experiments. The project started in 1984 and completed in 1989.

The main facilities of HESYRL are a 200 MeV electron linear accelerator, a beam transport line, and an 800 MeV electron storage ring. The control system are mainly divided into three relatively independent parts: a linac control system, a ring control system and a timing system.

1. System Configuration

The HESYRL ring control system is a distributed computer control system composed of a PDP11/45 computer, two 286 PCs, a communication microcomputer and up to 40 local control microcomputers(LCM). Fig. 4 is a block diagram of the ring control system.

Originally we plan to use a VAX11/780 as the main control computer. Both budget and delivery time problem led us to the choice of a PDP11/45 computer as a temporary alternative. Now because of the small memory size and poor display ability, its function is reduced to part of communication. Main control functions and console operation are all performed by the two PCs which are connected to serial ports of the PDP.

Ring equipments are controlled by local control microcomputers. Most of them are MULTIBUS system composed of a crate, a CPU board, a serial communication/memory expansion board and a several other interface boards. Because PCs are more convenient in programming and display we have also used some PCs as LCMs. The LCMs are located near the equipments. One LCM controls only one or one type of equipments.

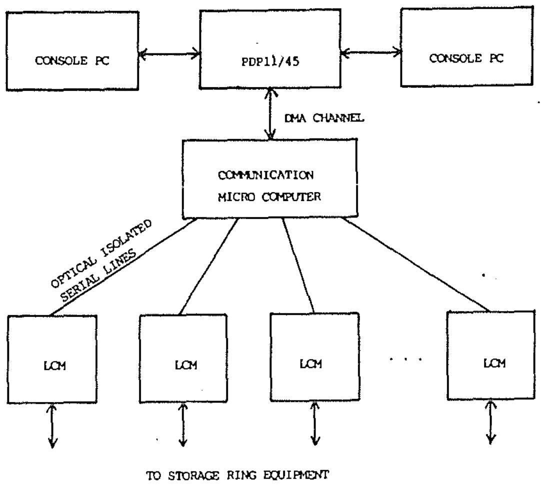


FIG. 4 BLOCK DIAGRAM OF HESYRL RING CONTROL SYSTEM

Because of the local intelligence of the LCMs, direct control and monitoring of the hardware equipments from the main computer are unnecessary. Most of operations are performed by the LCMs, system reliability and speed is improved. This is more obvious for ramping operation. With preloaded ramping table and hardware synchronization, ramping time can be less than 1 minute.

The CMM is a microcomputer specially designed for communication. It is a MULTIBUS system consists of a master CPU board, a DMA communication board and up to 10 intelligent serial communication board. CMM exchanges data with PDP through DMA channel and communicates with LCMs through serial links. Maximum number of channels is 40.

Low cost and convenient optical isolated asynchronous serial lines are used for data communication between the CMM and LCMs.

2. Software

The software for the control system consists of three levels: ROM monitor for the LCMs, communication program and ring control program.

ROM monitor of the LCM micros manages the LCM resources, handles communication and performs various control and monitoring functions.

The communication program includes ROM control monitor for the CMM, DR11-B driver and data buffer manager for the PDP.

The main function of CMM monitor is to handle the data exchange between the LCMs and the PDP. All data are exchanged in records. Error detection and retransmission are implemented.

The PDP DR11-B driver is installed as a device driver under the RSX11-M operating system and can be invoked by QIO. LCM status informations are stored in a shared data region on the PDP which is accessible by the console PC programs.

Ring control program includes transport line control, ring device status display, ring lattice setting and changing, beam energy ramping process control, close orbit correction and equipment testing programs.

Presently only transport line control, lattice setting, ramping control program and orbit correction programs are executed on the console PCs. Ring modeling, ramping table generation and other calculations are performed off-line.

3. Present status

The system was completed in March 1989 and has been operated since then. During the commissioning period the control system has performed satisfactorily.

Ring magnet control assured stable ring operation.

Ring control programs serve well for injection and ramping process control. But off-line calculation of physics parameters is inconvenient for machine study.

Beam energy ramping control is very successful. With the table ramping technique energy ramping or transfer to different lattice is very easy and smooth. Beam loss rate from 200 MeV to 800 MeV with 200 mA current is less than 7%. This is a good indication of ramping accuracy.

System communication has been reliable. But the response is slow. This is due to the speed of serial lines between console PCs and the PDP. LCMs are reliable and easy to maintain.

4. Upgrading plan

Several problems exist in the present HESYRL control system: 1) Transport line control and vacuum monitoring are now controlled by two standalone PC control system. They are not linked to the ring control system. This is not convenient for system management and data logging. 2) PDP seems unnecessary in the communication system, also it becomes a weak point. 3) Lack of on-line calculation ability is not desirable.

In order to solve the above problems, system improvements are planned.

DECNET will be implemented to link a Micro VAXII, the console PCs, transport line control PC, vacuum monitoring PC and a recently installed VAX6310 system together. This will give us a fully linked system with much expanded computing power.

The PDP computer will be replaced by two BIT3 bus connector cards which connect PC bus to MULTIBUS of the communication microcomputer. This will eliminate the bottle-neck and increase data speed of the whole system.

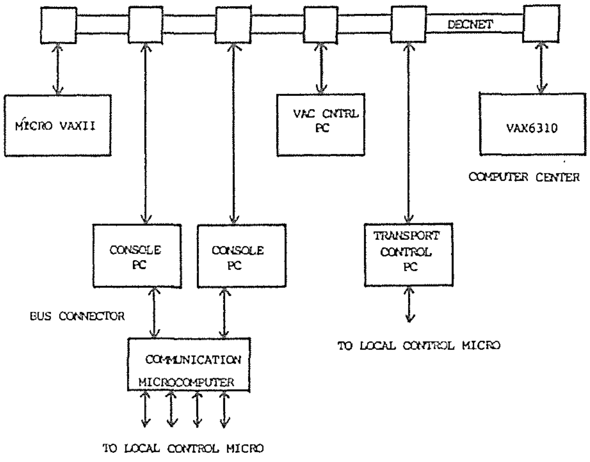


FIG.5 UPGRADE OF HESYRL CONTROL SYSTEM

Console display and computation power will be upgraded by replacing the console PCs with more powerful 386 or 486 stations.

Control programs and system communication programs will be rewritten to include on-line calculation function and node to node communication.

V. CONCLUSION

Started almost in same period, the control systems of the three accelerators are now all in operation. In parallel with the development of the systems, our chinese colleagues have gained experiences and established their qualified cooperative teams. This is a promising start.

Compared with world advanced laboratories our systems are not advanced no matter in architecture, hardware or software. Some system upgrading and optimizing work remains to be carried out.

VI. ACKNOWLEDGMENT

During design and construction periods we have received great help from SLAC, NSLS, KEK, CERN and many other laboratories. we would like to take this opportunity to express our thanks.

I would like to thank Prof.S.Y. Liu, D.K. Liu of IHEP and Prof. T.S. Jiao of IMP for helpful discussions and letter communications in preparing this report.

VII. REFERENCES

1. BEPC control group, The Status and Upgrade of BEPC Control System, BEPC workshop, June 1991, Beijing
2. T.S.Jiao, HIRFL Control System, Proc. of The Fourth Symposium of PASC

HESYRL Control System Status

Yao Chih-Yuan
Hefei National Synchrotron Radiation Laboratory
U.S.T.C.

Abstract

HESYRL synchrotron radiation storage ring was completed in 1989 and has been in commissioning since then. Now it has met its design specification and is ready for synchrotron light experiments. Control system of the project was completed in 1989 and some modifications were made during commissioning. This paper describes its present configuration, status and upgrading plan.

I. INTRODUCTION

Hefei National Synchrotron Laboratory (HESYRL) is a dedicated synchrotron light source. It's main facilities are a 200 MeV electron linear accelerator, a beam transport line, an 800 MeV electron storage ring and the experiment stations.

Design and construction of the ring and linac control system started in Oct. 1984. The linac control system was completed in 1987 and the ring control system was completed in 1989. Modifications have been made to the system during two years of machine commissioning, including RF control, vacuum monitoring and new control programs.

The ring control system is a distributed computer control system consists of a PDP11/45 computer, two PCs, a communication microcomputer system(CMM) and up to 40 local control microcomputer systems(LCM).

The linac control is essentially a manual control system.

A timing system consisting of two microcomputers provides all necessary triggering signals for the linac and the injection system.

II. RING CONTROL SYSTEM

A. System Configuration

Fig.1 is a block diagram of the ring control system.

Two PCs perform the main control function. Console display and command input are also performed on the PCs. The PDP11/45 minicomputer system, originally planned to be employed as the main control computer, is now only a part of communication system because of its limited memory and poor display ability. The two PCs are connected to the DZ11 ports of the PDP. Ring control programs are executed on the PCs.

Storage ring and beam transport line equipments are controlled and monitored by local control micros. The LCMs are located near the equipments. Each LCM controls only one or one type of equipments.

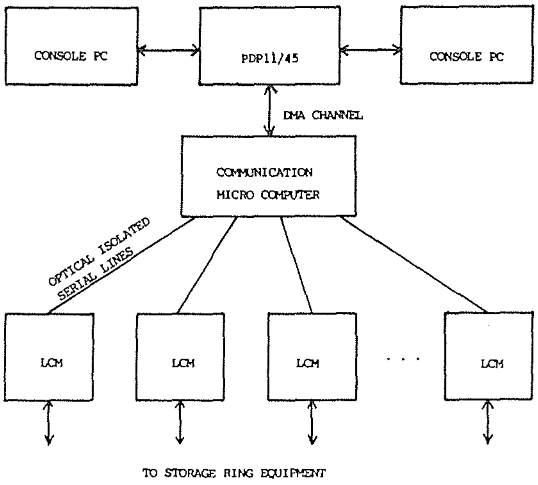


FIG. 1 BLOCK DIAGRAM OF HESYRL RING CONTROL SYSTEM

A LCM consists of a MULTIBUS crate, a SBC80/24 or SBC80/20-4 CPU board, a home designed serial interface and memory expansion board(HCOM) for communication and a few interface boards for equipment interfacing.

System communication is performed at two levels. The console PCs exchange data and commands with PDP through its serial lines. The PDP communicates with LCMs via a dedicated communication microcomputer system, which is a MULTIBUS system composed of a master SBC 80/24 CPU board, a DMA communication board and between 1 to 10 intelligent communication boards(COMM).

The COMM board is similar to Intel SBC544 intelligent asynchronous communication board. It has a Z80 CPU, 4 serial ports, on board RAM/ROM and dual port RAM memory which can be accessed by both a MULTIBUS master and the on board Z80. One COMM board can handle communication with 4 LCMs.

The DMA board controls DMA data transfer between the dual port memory and a DR11-B DMA interface of the PDP.

The main control console is made up of 9 modified industrial cabinets which are equipped with terminals, video monitors and various control panels. It is functionally partitioned into several subpanels: transport line magnet control, injection system control, timing control, ring control, beam diagnosis, interlock and personal safety etc. Most of the operations are performed at the console PCs.

B. System Software

There are three levels of control programs.

ROM monitors of the LCMs are at the lowest level. The basic design idea of LCM monitor is brought from NSLS. It consists of two parts: a basic control monitor and an application part. The former initializes the micro, handles communication messages and schedules operation of different control processes. The latter is essentially a collection of interfacing routines designed for the particular hardware configuration. The Basic control monitor is identical for all the LCMs.

At the second level are communication programs including ROM control monitors for the communication microcomputer, and a DR11-B I/O driver for the PDP.

The main function of SBC80/24 monitor is to handle the data exchange between the dual port memory and the PDP. All data are transferred in DMA mode.

There is a ROM monitor program on each COMM board of the CMM, which manages communication with LCMs.

The I/O program for DR11-B interface is installed as a RSX11-M device driver which can be called by QIO.

Ring and transportline control programs are at the third level. Three programs are used for normal operation. A program called RINGTEST performs ring device status display, ring magnet current and RF voltage setting, console command interpretation, ring lattice adjustment, file saving and restoration, beam energy ramping process control etc. It mainly deals with hardware signals. The second is BLOS, which performs transport line and linac magnet setting and adjustment. The third program, CORRECT, does closed orbit correction.

Ring lattice modeling, ramping table generation, closed orbit analysis and other calculation are performed by off-line programs.

C. Equipment Interfacing

The LCMs are mainly partitioned into the following seven subsystems:

HESYRL ring has 1 dipole group, 8 quadrupole groups and two sextupole groups. The main magnet power supply control system consists of 12 LCMs, 11 of them are ramping magnet LCMs each controls one magnet power supply. A LSI-11 controlled CAMAC system and a DVM is used to read back DCCT values of all the magnet currents. The dipole LCM also contains a ramping timer board which is a synchronizer for all other LCMs. Fig. 2 shows the configuration of the main magnet control system.

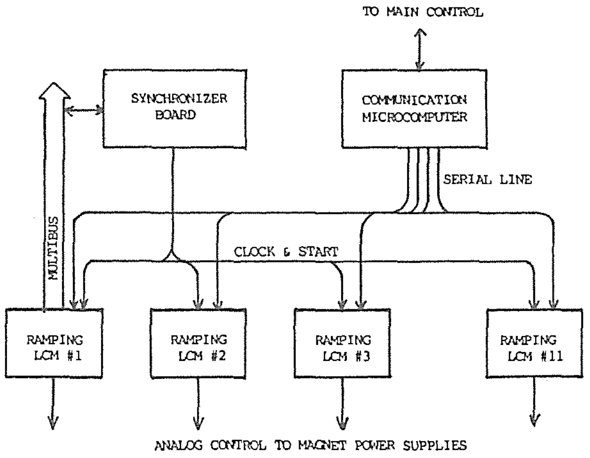


FIG.2 BLOCK DIAGRAM OF RAMPING CONTROL SYSTEM

The injection energy of HESYRL ring is 200 MeV, which is only 1/4 of its operation energy of 800 MeV. Ramping process is critical for high beam current.

During an energy ramping or ring configuration change process the currents of the magnet power supplies must follow certain calculated curve synchronously in order to keep the beam stable. In a centralized system the central computer has to modify magnet current one by one in many small steps. It will takes a lot of CPU time. The local intelligence of LCM allow us to take different approach. It is called table ramping. Ramping data are pre-stored to the LCMs. Synchronization of different LCMs is kept by hardware signals.

A ramping magnet LCM is composed of a CPU, a serial line interface, a ramping controller board and a 16 bit D/A converter board.

A ramping curve is divided into many small straight segments. Each segment is defined by its end current and slope values. As many as 1000 segments can be defined for one curve. A ramping segment is generated by counting up or down of a 16 bit counter on the ramping controller. The clock input to the counter is divided from a common clock output signal of the synchronizer. The ramping LCMs are programmed such that when a segment end is reached the new segment and slope values are loaded. The synchronizer also provides a start signal for all the ramping LCMs. Tracking of different magnets is automatically assured.

In order to start a ramping process, the main computer just needs to load the segment table, send necessary command to the LCMs and then issue a start command to the dipole LCM to start the synchronizer.

A fitting program is developed to calculate optimized ramping table for any required ramping process. The theoretical fitting error is 1 bit.

12 auxiliary windings of dipole magnets and 64 auxiliary windings of quadrupoles are configured as 12 horizontal and 32 vertical closed orbit correctors. Two LCMs are used to control all these corrector power supplies.

Control of transportline focusing and correction magnets are performed by system composed of a SBC80/24 LCM in the power supply room and a PC on the operator console. The PC acts as the control master and the LCM accepts PC command and performs the actual control.

The beam position monitor signals are needed on line for closed orbit correction. A PC data acquisition system is used to read and display the BPM data.

Ion pump currents and vacuum gauge pressure readings are monitored by a PC data acquisition system.

Injection system includes a pulsed septum magnet and three fast kickers. The triggering signals are provided by timing system. Pulse amplitude control are performed by a SBC80/20-4 LCM with a terminal on the operator console for command input.

HESYRL ring has only one RF cavity, so phase control is not necessary. A tuning loop is built for cavity tuning control. Detuning angle is controlled by a manual adjusted phase shifter. Cavity voltage is regulated by a feedback loop. A ramping LCM is used to control the cavity voltage. During ramping process the cavity voltage is controlled in the same way as the ring magnets and therefore can follow any desired curve.

III. LINAC CONTROL SYSTEM

The linac control system is divided into 8 separate subsystems each has its own control panels and controls one part of the linac equipment. The modulator control panels, klystron focusing control panels, electron gun control panels, microwave control panels, vacuum control panel are all manual control panels.

The 200 MeV linac has an extension section for future beam energy expansion. Linac focusing and correction magnets fall into two groups: 30 focusing and steering magnets for linac proper and 32 magnets for the extension section including a beam energy analyzer. They are controlled by two SBC80/20-4 LCMs and a PC. The PC serves as the master and operator console. The LCMs perform the local control and monitoring functions. The links between the PC and the two LCMs are fiber optical serial lines.

A SIEMENS S5-101 programmable controller is used for linac interlock and protection.

IV. TIMING SYSTEM

The timing system provides synchronization triggering signals for the linac, injection system and beam diagnose system. Since timing is only adjusted during injection and linac set up time and most of its parameters are fixed, It is built as an independent system. Two LCMs are implemented: a main timing micro(MTM) installed in the main control room and a linac timing micro(LTM) in the linac control room.

The MTM generates master start and clock signals which are the reference of the whole system. The master signals are transmitted to the linac timing micro. MTM also provides trigger signals for the septum, kickers and for beam diagnose system. The LTM generates trigger signals for the electron gun, the microwave source, the klystron modulators and the switching magnet pulse generator.

Structurally the two micros are almost identical. They consists of a CPU board and a timing generation board(TIG).

Present timing system can only be operated in full bunch injection mode.

V. PRESENT STATUS

During the commissioning period the control system has performed satisfactorily.

Timing system, linac and transport line magnet control subsystems greatly eased linac and transportline tuning. Resolution and delay adjust range of timing system are suitable for optimization of linac operation.

Injection control and ring timing systems assured conveniently setting of correct injection condition. The ring control program has a linear approximation trimming function to adjust the ring energy slightly up or down. This is very useful for matching ring and injection beam energy to improve injection rate.

The measured accuracy of ring magnet current setting is better than $7X10^{-5}$. This assured stable and repeatable ring lattice setting. The measured tune variation is less than .001.

The simple practical approach of RINGTEST control program served well in

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S01SRA10

machine commissioning and normal operation.

Beam energy ramping control is very successful. With the table ramping technique energy ramping or transfer to different lattice is very easy and smooth. Current tracking was measured better than 10^{-4}. Beam loss rate from 200 MeV to 800 MeV with 200 mA current is less than 7%, which is of same order as normal lifetime loss. This is also a good indication of ramping accuracy.

System communication has been reliable, but the system response is slow. This is due to the speed of serial lines between console PCs and the PDP.

VI. PLANNED SYSTEM UPGRADE

The operation experience showed us that there exist several problems in the present HESYRL control system: 1) It is not a fully linked system. A few standalone subsystems are not connected. This is not convenient for system management and systematic data logging. 2) PDP is an old system and difficult to maintain. It becomes a weak point. 3) Lack of on-line calculation ability is not desirable.

Having analyzed these problems, and considering our available resources, we plan to make the following improvements:

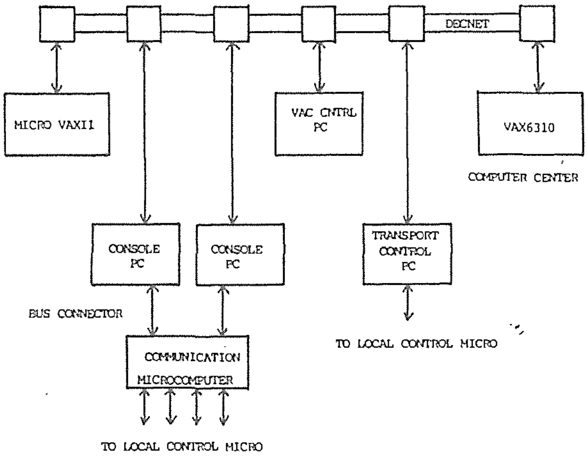


FIG.3 UPGRADE OF HESYRL CONTROL SYSTEM

1) Introducing DECNET to link the MicroVAXII, console PCs, transportline PC, vacuum monitoring PC and a recently installed VAX6310 system together. This will give us a fully linked system and much expanded computing power.

2) Replacing the PDP by two BIT3 bus connector cards which connect PC bus to MULTIBUS of communication system and directly map the dual port memory to PC memory. This will eliminate the bottleneck and increase data speed of the whole system

3) Improving console display and computation ability by replacing the console PC with more powerful 386 or 486 stations.

4) CATV is very convenient to display machine status information to the users and other laboratory staffs. Installation of a 10 channel CATV system is under way.

5) Control programs will be rewritten to include on-line calculation function and node to node communication.

VII. ACKNOWLEDGMENTS

J.Yang of SSRL, J. Sheehan and J. Smith of NSLS contributed their suggestions during their visit to our laboratory.

The control group have been working cooperatively over the years to build and maintain the system.

VIII REFERENCES

1. S. Magyary et al, Operating Experience with a New Accelerator Control system Based upon Microprocessors, Proc. of 1981 Particle Accelerator Conference, Washington D.C.

2. K. Batchlor et al, Distributed Control System for the National Synchrotron Light Source, Proc. of 1979 Particle Accelerator Conference, San Francisco

3. Yao Chihyuan, Beam Control and Diagnostics of HESYRL Ring, Proc. of the 4th Japan-China Symposium on accelerator and its application, 1990, Beijing

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S01SRA11

The Control System of HIRFL

Jiao Tianshu, Li Tianyou,Ma Siwen,
Chu Zhensheng, Huang Tuanhua, Zhou Xun,
Wang Zhen,Shen Zhiqing
Institute of Modern Physics, Accadmia, Sinica
No.57, Nanchang Road, Lanzhou

1. Introduction

The Heavy Ion Research Facility in Lanzhou (HIRFL) is a multi–purpose and variable energy machine designed to accelerate wide range of ions. (1) In order to obtain a designed beam (particale and energy) and to transport it to a proper experimental areas in a short time, it requires to modify a great number of parameters, this cannot be easily achieved without the help of a computer.

The control system design and construction was started in 1983. First of all, some local control station of accelerator subsystems were finished in 1988 and satisfied the needs of operating and commissioning at the elementary level. Controlling the HIRFL process is implementing at a high level.

2. The brief description of control system

Fig.1 shows the general layout of the control system for HIRFL.(2) It is based on CAMAC distributed process configuration.(3)

(1) The local computer control stations are designed according to the accerlerator subsystems, such as Magnet, R.F, Vacuum, Injection and Extraction, Beam line etc. and were finished in 1988. They can meet the case of beam tuning and accelerator operating at elementary level.

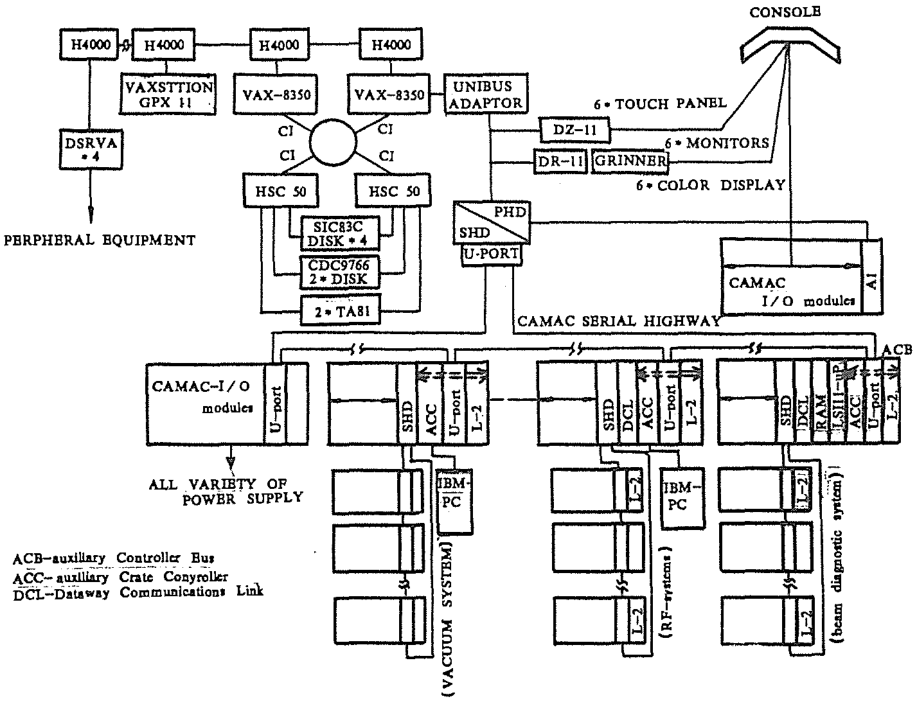


Fig.1 The block diagram of HIRFL control system

(2) All local stations microcomputer links to Host computer by CAMAC serial loop line. Two communication machanism are available:

(1) A memory "mailbox" CAMAC module accessible from both host and remote processors. (2) A CAMAC Dataway Communications module or "DMA port" between the remote processor memory and the Dataway.

2.1 Computer

HIRFL is controled by means of 2 kinds of processors: mini–computers as host computer and microcomputer as local stations. They are linked with CAMAC serial loop line.

The host computer is two VAX–8350 that are connected by cluster configuration. They share 4x520MB disks, 2x300MB movable disks and tapes. Each computer equipped with 12MB memory. One of them is used as a host computer for HIRFL control system. The other is used as a reserved computer when the former one is in fault. In addition, it is also used calculation and data processing for the experiments carried out in the experimental areas.

Microcomputers for the local stations are LSI–11/23 and IBM–PC. They are used to control subsystem for Vacuum, R.F, Beam diagnosis system and so on. These microcomputers are a completed system with memory, disk, terminal.

VAX–8350 control the SSC Cyclotron through the CAMAC bit serial loop. This loop was drived by serial highway driver. Serial rate is 2.5MB. There are about 20 CAMAC crates, corresponding CAMAC module are used for the interface of devices.

2.2 Device control

In order to bring into being link and matching between CAMAC modules and devices, 5V digitalized I/O–signals, 24V switch signals, and 5V analog voltage signals was planned. A variety of condition circuits, such as switich board, status board, DAC adjusting board with accuracy of 16bit–18bit and ADC data acquisition board with accuracy of 12–16bit, had been designed and made. They are used for on/off power supply, status monitor, current adjusting and data acquisition.

A step motor control is used for units requiring accurate position setting, such as electrostatic deflectors, magnetic channels. The controller of step motor is designed to provide with local control function. They are slow acceleration when movable units is started and slow deceleration when movable units is derived to the front of end. So movable units not only can keep running smoothly, but also can obtain higher operating speed.

In order to save funds, a number of digital and analogue multiplexer is used for devices of some slower action. Otherwise, the pressure operated control is mainly used for the units of two movable position such as Faraday cup, vaccum value, secondary emisson multi–wire profile monitor etc.

3. *Console*

Our console is designed after a careful analysis of which have been done in GANAL[(4)] and RIKEN[(5)].

The console, mechanically built with 10 benches is devided in 3 operating console units L, C, R. The central console unit C is devoted to equipments not linked to the computer, such as worksite monitor, RF waveform and beam signal obeservation. The console units L and R are identical.

Controlling the HIRFL process with the console system can be exercised at the elementary level or a higher level. At the elementary level the control system behaves as a large multiplexer. Equipments designed by their device name are handled one by one.

The operator uses the touch panel and the turn pages of the device name to choose one corresponding equipment he wants to control. Then, operator moves a cursor to a selected device name on TV screen with a defined touch key and reads the various informations such as the controlled value the actual value of the parameter and a status word on the TV screen area designed.

Among the numerous signals collected along the accelerator, most of them concerning parameters values or status are digitalized and enter the data flow transmitted by the CAMAC system for processing. A little analog form is remain which are to be used in their anolog form. This is the case of some beam diagnosis signals and R.F signals with time. These waveform siganl observation is necessary.

These signals can be displayed on autoranging picoammeters or on oscilloscopes.

4. The present situation

The HIRFL operating software called HIRFLCSF which is written in Fortran 77. Presently, center console programs are developping for the man–machine interface with Touch Panel. With time goes on, HIRFL operating level will improve.

Refernce

[1]. Zhang Enhou, Wei Baowen etc., Survey of Heavey Ion Research Facility in Lanzhou (HIRFL), proceedings of Heavy Ion Research Facility in Lanzhou, Vol. 8, 1 (1989)

[2]. Jiao Tianshu, The Control and Beam Diagnostic System, proceedings of Heavy Ion Research Facility in Lanzhou, Vol 8, 87 (1989)

[3]. J. W. Tippie, etc., Discussion of Tools and Techniques for Distributed Processor Based Control Systems Using CAMAC, IEEE Transactions for 1985

[4]. I. M. Prome, ″The GANIL Control System″, IEEE Transactions on Nuclear Science, Vol. NS–28, No.3, June 1981

[5]. T. Wada, H. Tarebe etc., Control system of RIKEN Ring Cyclotron, RIKEN Accelerator Progress Report, Vol.201, 170(1986)

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S01SRA12

Control System for a Heavy-Ion Accelerator Complex K4 – K10

V. M. Kotov and R. Pose
Joint Institute for Nuclear Research
Laboratory of Computing Techniques and Automation
Dubna, USSR

Abstract

Control systems for newly created accelerators, perhaps for the first time, may be designed almost only around international standards for communication and control techniques. This is also true for the project of a control system for the accelerator complex K4-K10 at the Joint Institute for Nuclear Research Dubna. Nevertheless, open systems architecture with construction principles being essential for modern systems of such big devices as particle accelerators leaves designers enough possibilities for solving even very sophisticated problems.

I. INTRODUCTION

The control system of the heavy-ion accelerator complex K4-K10 is similar to the systems developed for controlling the accelerators of other physical laboratories the experience of which was used in preparing the given project [1,2,3]. Nevertheless irrespective of the similarity of accelerators and control problems there are no equal control systems. This depends not only on the differences of accelerators as such but first of all on the time the system was designed and on the present state of computing and communication technique and on the special features of the system architecture, which allow new technical acquisitions during the realisation.

II. THE ARCHITECTURE OF THE K4-K10 CONTROL SYSTEM

The architecture of the K4-K10 control system is based on a two level distributed computing system (Fig. 1). The upper level uses a modular system IEEE 1296 (Multibus) from the Siemens AG [4] (SIMICRO) as well as workstations with UNIX and X Window in the Central Control Room, and is integrated into a system via a Local Area Network (IEEE 802.3).

The standard IEEE 1296 and its realisation in the SIMICRO products allows a high computing power inside every crate (the CPU on-board in the SC^{TM}-systems based on RISC processors providing 5 MIPS and more) as well as the use of CPU with embedded PC/AT 386 in the OSM^{TM} and AMS^{TM} systems[1]. These SIMICRO systems are equipped with a complete set of communication modules for LAN in the IEEE 802.XX standards.

The functions of the system at this level are supported by the operating system SORIX^{TM} - a UNIX^{TM} Real Time version (reaction time for interrupt signals ≤ 100 μsec.) and permit an exit to LAN via TCP/IP protocols[2].

Therefore, using the possibilities of the 7 level scheme for the OSI model of open systems, the architecture of the upper level of the control system provides an output to the equipment and to the real time programs as well as a complete support for the TCP/IP protocols for an output to a LAN IEEE 802.XX. According to the loading of the network at this level by means of bridges one may distinguish two LAN segments: one for the workstations WS in the Central Control Room running under UNIX and the other for the Front End Computing. The latter are mainly crates in the IEEE 1296 standard with Real Time UNIX, joining the node computers for data acquisition and handling in the control mode as well as that for graphics and monitoring. They are equipped with terminals, Touch Screens and other man-machine communication devices.

The lower level of the distributed computing system directly corresponding to the equipment and executing devices and based on standards for industrial systems (for example in the VEPP4 [5] CAMAC instrumentation is used at this level) is connected to the upper level by means of the communication environment FIELD BUS (MIL-1553-B). This standard mostly agrees with the demands for a multidrop bus and in 1983 was proposed as a standard protocol for the field bus in accelerator control systems [6]. Together with well developed hand shaking features for the message transfer between bus controllers and remote terminals (RT) this standard allows simple and cheap data transfer to single serving devices, fulfilling simplest functions. For connecting digital measuring devices to the upper level the standard IEEE 488.X is used.

For the most part all the bus controllers (BC) MIL-1553-B are placed on the upper level of the control system. They provide the interfacing of the node computing system to the lower level equipment. The BC for the K4-K10 control system is developed on the basis of a processor module in the Multibus II standard, designed in the Laboratory of Computing Technique and Automation of the JINR using the VLSI of a 32-bit microprocessor set K1839 software-compatible with a micro VAX [7]. The interface controller for the MIL-1553 bus is a piggy-back module for the CPU board. One such BC may control 30 standard interfaces (Remote Terminals) on each bus.

The standard interface of the MIL-1553 bus has to provide a prompt/reply regime. In this case a microprocessor is needed. In the case of simple messages of the type "adjust and/or read" it has to handle the transfer directly.

The use of the MIL-1553 for the communication with the equipment together with the time synchronization channel allows one to put the real time mode on the lowest control level. Such a solution proposed and realized in the GSI [1] relieves the upper control level of the real time business and allows on this level the use of the LAN 802.3 alone without a TOKEN RING (IEEE802.5) as it is done for the SPS/LEP in CERN [2].

---

[1] TM SIMICRO, SX, SORIX, OSM, ASM - Trade Marks of SIEMENS AG

[2] TM UNIX - Trade Mark of AT & T

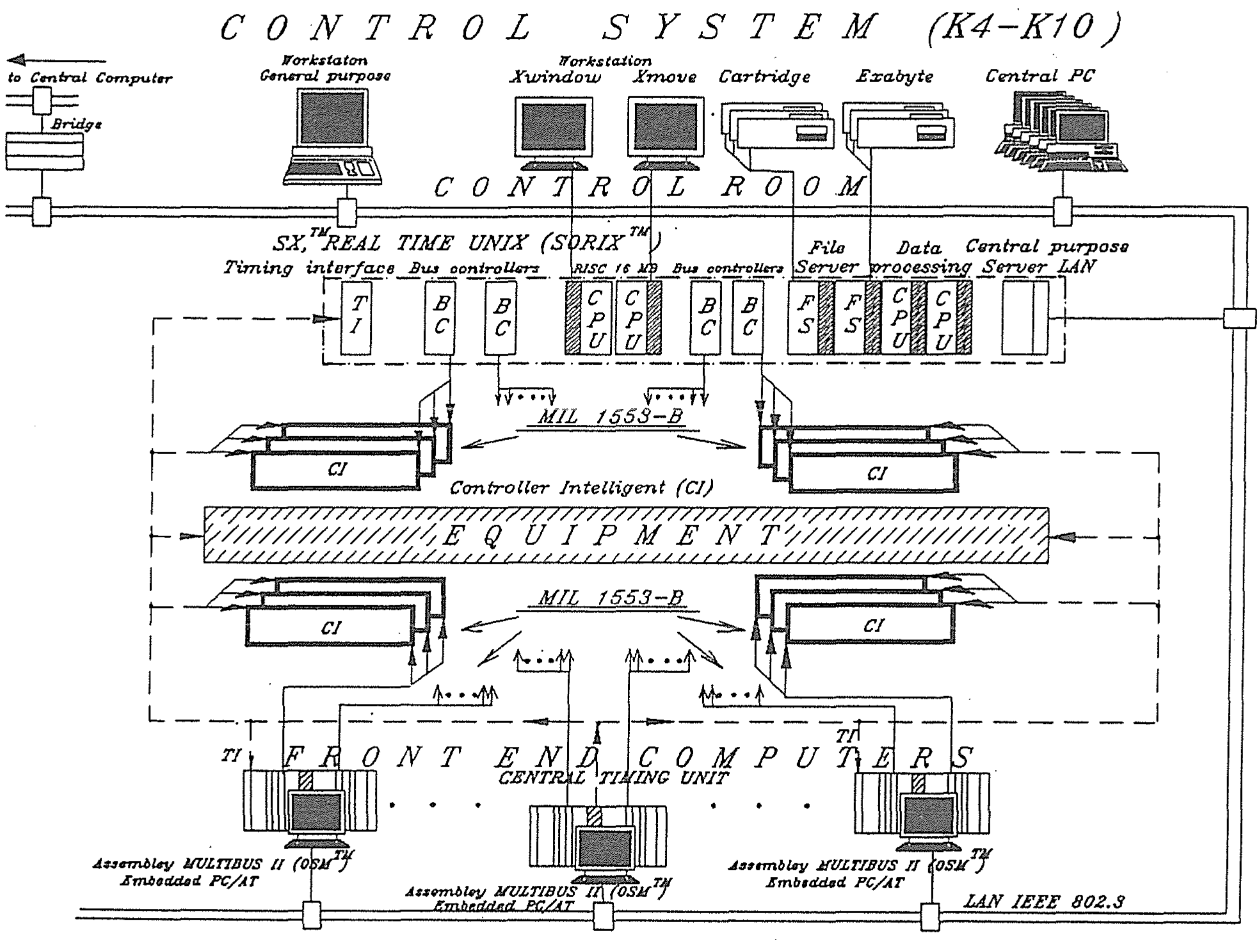


FIG. 1

real time operation and will be stored into the equipment controllers at the lower level.

In the project of the control system for the K4-K10 it is also planned to use such a kind of real time system for the lower control level. But taking into account relatively small dimensions of the accelerator rings of the K4-K10 (the perimeter of the K10 is about 200 m) and the transfer rate of 1 MByte/sec. at distances up to 300 m for the MIL-1553, the connection of the main node computers of the control system at the front-end level may be done without a LAN 802.5, directly on the system bus of the Multibus II with a transfer rate of 40 MByte/sec and full message passing mode. In such a way the possibilities of real time control are widened allowing for this purpose the use of interprocessor transfer in the Front-End Computing of the upper level.

The control system architecture of the K4-K10 allows the use of the standard operating system UNIX and SORIX for the node computers. The main programming languages are C at the system level and NODAL [8] at the application level. The software at the lower control level [2] has to be as short as possible (up to 10 KByte because at this level one may use 8 bit microprocessors) and has to provide a good reactivity: up to 250 interrupts per second with quite long messages on the MIL-1553 and up to 20 Kbyte/sec. All the software on the lower level is part of the system data base. It will be picked out together with all tables of parameter sets necessary for the

III. CONCLUSION

The control system architecture for the accelerator complex K4-K10 represents an example of an open architecture system based on the use of control and communication software and hardware corresponding to international standards. Though the system designers have the possibility to introduce their own original technical solutions, the main effort will be directed to the software and hardware design concerning the interfacing part of the system, i.e. the hardware and software ensuring the application of system resources as some kind of a set of control facilities.

REFERENCES

[1] R. Steiner et al.: The GSI Control System. Europhysics Conference on Control Systems for Experimental Physics. Villars-sur-Ollon, Switzerland, page 21-25, CERN 90-08, 12. 10. 1990.

[2] R. Rausch: Control System Architecture of the European Accelerators LEP and SPS. CERN SL/90-109 (CO).

[3] S.Kurokawa et al.: The Status and Prospects of the TRISTAN Controls System. Europhysics Conference on Control Systems, page 15-21, CERN 90-08, 12. 10. 1990.
[4] SIEMENS SIMICRO, Industrial Microcomputers SIMICRO SX. Catalog MC, 4. 1. 1991.
[5] A. N. Aleshov et al.: Accelerator Control System at the INP. Europhysics Conference on Control Systems for Experimental Physics, Villars-sur-Ollon, Switzerland, page 25-30, CERN 90-08, 12. 10. 90.
[6] D. Francart et al.: MIL-1553-B Multidrop Bus for Controlling LEP and SPS Equipment. LEP Control Note 56 rev.
[7] V. Kotov, L. Dubovic, R. pose: Automation and Control System for a cτ Factory. Proceedings Workshop cτ Factory in Dubna 1991.
[8] M. C. Crowley-Milling and G. C. Shering: The NODAL System for the SPS. CERN 78-07 SPS Div., 3. 9. 1978.

Future Directions in Controlling the LAMPF-PSR Accelerator Complex at Los Alamos National Laboratory*

R. Stuewe, S. Schaller, E. Bjorklund, M. Burns,
T. Callaway, G. Carr, S. Cohen, D. Kubicek,
M. Harrington, R. Poore, and D. Schultz
Los Alamos National Laboratory
Los Alamos, NM USA 87545

Abstract

Four interrelated projects are underway whose purpose is to migrate the LAMPF-PSR Accelerator Complex control systems to a system with a common set of hardware and software components. Project goals address problems in performance, maintenance and growth potential. Front-end hardware, operator interface hardware and software, computer systems, network systems and data system software are being simultaneously upgraded as part of these efforts. The efforts are being coordinated to provide for a smooth and timely migration to a client-server model-based data acquisition and control system. An increased use of distributed intelligence at both the front-end and the operator interface is a key element of the projects.*

I. INTRODUCTION

The integration of the Los Alamos Meson Physics Facility (LAMPF) and the Proton Storage Ring (PSR) control systems is presenting a series of problems for the operations and support personnel using the two systems. The two systems were developed independently using different personnel, different underlying philosophies and different equipment but developed interdependency when the operating and support groups were combined in 1988. A detailed discussion of the current control systems is presented in a companion paper in these proceedings.

II. PROBLEMS AND IMPACT

A. *LAMPF RICE System*

The LAMPF Control System (LCS) was built upon the LAMPF-designed Remote Instrumentation and Control Equipment (RICE) System. RICE is the hardware and software interface between the actual accelerator devices, such as magnets and beam-line instrumentation, and the software that operators and developers use to control beams.This system is illustrated in Figure 1. RICE presently utilizes 73 of 80 possible modules handling 10,000 data and control points distributed along approximately 2 km of beam channels.

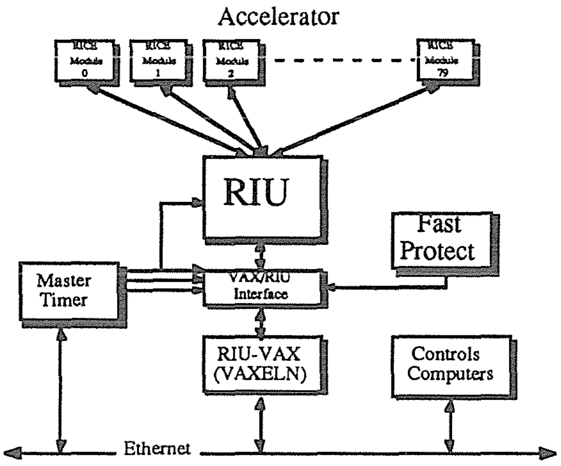


Figure 1. The LAMPF RICE System.

The RICE system has several limitations. First, RICE allows only one timed data request per beam pulse. At the low repetition rates characteristic of tuning beams, this feature can cause requests for device readout to become deeply queued and can cause tuning mistakes as operators react to "old" data. Second, the RICE star architecture places limits on the maximum data rates. Third, the RICE system is fairly rigid. For example, the current implementation of the linac harp system permits data from only two harps at any time and requires availability of greater than 50% of the RICE system to obtain that data. Fourth, there is a need for higher accuracy and precision than is available with RICE. Finally, it is estimated that 30% of the parts in the RICE system are no longer commercially available.

B. *PSR ISS System*

The PSR Control System is based upon a series of PDP-11 computers known as Instrumentation Sub-Systems (ISSes) which communicate with a central VAX system using serial CAMAC [1]. This is illustrated in Figure 2.

---

* Work supported by the US Department of Energy

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S02SRU01

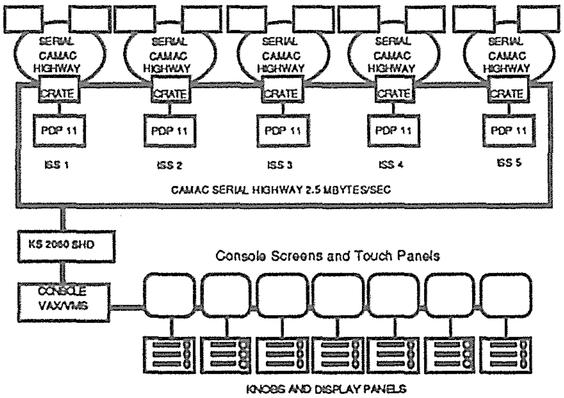


Figure 2. The PSR Control System.

The most significant difficulty with the PSR Control System is that the current update rate is one-fifth of what is considered optimal. This rate is directly limited by the system architecture which migrates all data continuously through the control system CAMAC loops to maintain the system's real-time database. Additionally, the complicated and time-consuming procedures required to modify the PSR database and the PDP-11 software discourage changes during run cycles, restricting the flexibility to correct problems and add devices. Finally, the PDP-11 database size limits are being approached on all five computers now that the system is twice its original design size.

C. LAMPF and PSR Operator Interface Systems

In the LAMPF control system, data display and entry is handled using color CRTs, graphics scopes, button panels, terminals, trackballs and knob panels [1]. Experience over the last two years has demonstrated that the demands for cpu power and data generated by consoles can easily outstrip any central control computer's ability to service the demand. This indicates that additional computing power is required to support future console demands.In the PSR control system, data display and entry is accomplished using color graphics screens, touch panels, and knob panels. The color graphics systems are creating an increasing maintenance burden due to difficulties in obtaining parts and adequate support from the manufacturer. The PSR touch panels are an inefficient and often ineffective method for interfacing to the control system.The graphics software of both systems is created at a fairly primitive level. This prohibits quick prototyping of application codes and increases the overall time required to create an application program.

III. GOALS AND CONSIDERATIONS

A. Performance Goals

Goals have been established for the performance elements capacity and operator console response time. In terms of capacity, the upgrades should provide the needed capacity to support twice the number of existing consoles and twice the number of data and control points in the system and increase the overall data throughput by a factor of three to 450 untimed read requests per second. Response time should be improved to provide a 5 Hz update rate at the operator consoles during normal operation and human speed, 1-2 Hz, beam profile information.

B. Availability and Maintainability Goals

Availability is a measure of the combination of failure rate and repair rate. Maintainability is the level of resources required to achieve a given availability. The availability goal for the upgrades is specified as 99.7%. This represents an improvement over the current systems which operate at approximately 99.4%. For these purposes the system is said to be unavailable if a failure occurs that prohibits planned beam tuning, development or production. To achieve this overall availability, the Mean Time To Failure (MTTF) goal is specified as 1 week, up from the current 4.2 days, and the Mean Time To Repair (MTTR) goal is specified as 0.5 hours, reduced from the current 1.3 hours. Achievement of these goals would provide approximately 11 hours additional beam time per typical annual operation period.

In order to support the capacity goals, the upgrades are being designed with maintainability in mind. Currently 9.5 software personnel support 21 control computers accessing approximately 16,000 data and control points. The upgrades specify as a goal that the current staffing level must be able to support a minimum of 40 front-end computers accessing up to 32,000 data and control points. To achieve this, the number of hardware diagnostics, software diagnostics and software tools must increase by at least a factor of two and the number of distinctly different hardware systems utilized must be similarly be reduced by a factor of two.

C. Considerations

Several specific considerations are being folded into the upgrade efforts. The first is long-term growth potential. This is directing effort toward the use of stable vendors and recognized and developing software and hardware standards. A desire for flexibility and robustness is to be addressed through increased modularity in both software and hardware. It would also be desirable if the systems were designed such that beam line developers could rapidly prototype and develop their own application software to reduce the support required by controls staff.

A major consideration of the proposed plan is the desire to migrate from proprietary systems, languages, and communication protocols to non-proprietary "open system" standards in all areas. For operating systems, the intent is to migrate to the POSIX open system standard when it is implemented. For communications, the Open System Interconnection (OSI) standard DECNET Phase V will be preferred if it meets performance needs. Portable languages such as FORTRAN, C and C++ will be preferred.

IV. THE SOLUTION MODEL

To address the problems a simplified model of the control systems was used. This model, shown in Figure 3, is a statement of the elements and interfaces which must be standardized if the desired results are to be obtained. The model illustrates four levels; Data Acquisition and Control Computers, Communication Systems, Applications Systems and Operator Interface Systems.

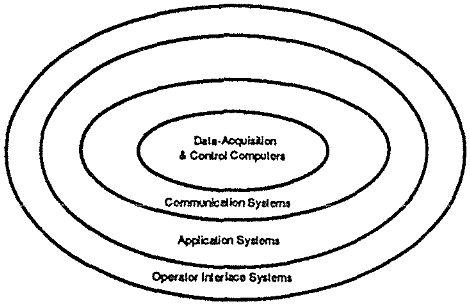


Figure 2. The Solution Model

A. Data Acquisition and control computers

This element of the model refers to the remote computer, hardware interface and low level software used to connect the front-end hardware to the communication medium.

B. Communication Systems

This element represents the hardware and software support required for a network based distributed computer control system. It provides the necessary interface between application software and the remote computers that actually acquire data and manipulate control points.

C. Application Systems

This element represents the algorithms and software systems that relate operator interface tools and beam line elements in a functionally useful manner. Physically, this element is an interface between the operator interface tools and the communication system.

D. Operator Interface Systems

This is the set of hardware and software mechanisms established to provide the user with the ability to interact with application software and thus the beam line instrumentation.

V. THE SOLUTION IMPLEMENTATION

The application systems, communication systems and remote data acquisition and control computers are already established in the form of existing LCS elements. The additional solutions developed must integrate well with both the existing and developing systems to preserve maintainability goals.

A. Data Acquisition and Control Computers

This element will be provided by the proven LCS VAXELN-CAMAC system. It is intended that this element provide both a RICE replacement and the PSR PDP-11 computer replacement.

VAXELN is the preferred remote computer operating system for the upgrade at this time. It provides a large array of development tools, integrates well with the current system and has an apparently solid upgrade path for the next 5-plus years. This implies a commitment to VMS, the development system for VAXELN, for at least this period. The vendor has indicated that both VMS and VAXELN will be POSIX compliant in the future, providing open system compatibility.

As stated, the device interface will be CAMAC, currently in use in the PSR system and in a significant portion of the LAMPF control system. As the PDP-11 computers are replaced, existing applications software will be modified to use a data-on-request approach thereby eliminating the resident PSR database and the current PSR control system over time. As RICE modules are replaced, the hardware that is removed will be used as spare parts to support the remaining RICE modules.

Experience with VAXELN controlled CAMAC systems provides confidence that the performance goals of the upgrade can be met. VAXELN/CAMAC driver testing has determined that CAMAC reads can be accomplished in 35-50 microseconds. A small number of these systems distributed throughout the site can meet the current and predicted data acquisition requirements.

B. Communication Systems

The communication system is intimately tied into the performance question. The LCS Remote Procedure Call (RPC) System will be utilized as the basic communication mechanism. Controls staff have performed, and are continuing to design, network performance tests to evaluate the system.

The level of network traffic anticipated for the new architecture has been estimated through measurement of peak demands on the current network with allowance for future expansion. Tests indicate that the current Ethernet/DECnet network is adequate for the near-term future even with the addition of the 10-20 computer nodes predicted for the RICE and PSR Upgrades. The effects of the 5-10 additional nodes needed to support proposed linac upgrades are currently being considered. An Ethernet/DECnet-based network provides a standard that is compatible with a variety of computer platforms and operating systems.

While a simple network architecture will accommodate current projections, future requirements for segmentation must be included in planning. The addition of increasingly powerful computers to the network is the most likely future growth path and problem. When the networks are no longer cpu power limited, segmentation, through the use of bridges and routers, and new network technology such as FDDI will be required to provide solutions.

C. Application Systems

An effective application system is dependent upon the standardization of certain sub-elements and the development of a consistent "application viewpoint" of the machine.

The manner in which an application views its relationship to the data it desires is a critical element. The current PSR applications view data as continuously available since the system uses a continuous polling mechanism. This contrasts with the data-on-demand LCS applications. The upgrades recognize that a system that combines polled-data at the remote level with demand-data at the application level may be required to provide the performance that is desired and adapt to the conflicting application viewpoints. A version of such a system is being used in the RIU-MicroVAX [1]. This system continuously polls frequently requested data so that values can be supplied from a real-time database for this data. Less frequently requested data is read from the hardware when demanded.

Questions surrounding application requirements for device control locks, access to global data, data integrity and error handling remain to be addressed. These issues are further complicated in the highly distributed system that is envisioned.

Concurrent with standardization of the applications systems, the standards that are in place for control system software development will be re-examined These include requirement, design and user documentation procedures as well as configuration management systems.

D. Operator Interface Systems

It is believed that VAX-based workstations will provide the common interface hardware to replace both PSR and LAMPF console systems. The selection of VMS systems is to provide compatibility with the current VMS-based application and system software. Efforts are underway to evaluate user interface management systems against operator and developer preferences. The rapid development of commercial software tools in this area promises to assist in this effort. The first step in this process is a planned emulation of the existing LAMPF color CRT. This will provide a common interface for all parts of the installation and provide computing power to off-load the central control computer, thereby improving performance. This distributed interface system will similarly contribute to an increase in control system availability by reducing the number of potential single points of failure.

VI. SUMMARY OF CURRENT EFFORTS

Several projects have been established to provide the upgrades to the systems as they have been described. The projects are the RICE Upgrade, the PSR ISS Upgrade, the Operator Interface Upgrade and the LAMPF Database Upgrade. Obviously, the pieces are not separate and must interact closely with each other to provide the standard elements of the proposed model. Existing resources and prioritization of the various problems are driving forces in selection of the solutions. Effort is already underway through work on all of the projects at various levels.

The RICE Upgrade has completed the requirements phase and is moving into design. Network and driver testing has been performed to insure that the systems selected will be capable of meeting the performance requirements. Evaluation of computer platforms is underway. The initial steps in defining a diagnostic system that will adequately support the new data system have been taken. Part of this effort is to design a CAMAC-based timed data system to replace the functionality of the RICE system.

The PSR ISS Upgrade is in the requirements phase, working rapidly in order to provide feedback to the RICE Upgrade project to insure that incompatibilities are not designed into the new front-end system. Efforts at defining the scope and complexity of the project are ongoing.

The Operator Interface Upgrade is also in the requirements phase. Much preliminary work has been performed. This work involved evaluating commercial Graphical User Interface's (GUI's), evaluating X-Windows as a graphics tool, determining what solutions are being used at other accelerator facilities and defining LAMPF control system user preferences.

The LAMPF Database Upgrade is in the requirements stage, although significant effort has been expended to bring the LAMPF database philosophically closer to what the PSR system requires. The LAMPF database can now support PSR devices, a necessary first step to integrating the systems [1].

A steering committee within the controls section has been established to coordinate the efforts. Quarterly internal project reviews and an external review of the combined projects are used as a means assuring the quality of the effort. The goals described in this document will be used as a measure of successful project completion.

VII. REFERENCES

[1] S. Schaller, R. Stuewe, et. al.,"Experience Controlling the LAMPF-PSR Accelerator Complex," these proceedings.

[2] G. Carr, S. Schaller, et al., "The Status of the LAMPF Control System Upgrade," Proc. Europhysics Conference on Control

Common Control System for the CERN accelerators

R. Rausch, Ch. Serre, editors
CERN - 1211 Geneva 23 - CH

Abstract

The PS and SPS Accelerator Control Systems are becoming obsolete and need urgent rejuvenation. After a control users forum, where users expressed their needs, two main Working Groups were set up, consisting of Control and Equipment Specialists and experienced Machine Operators. One Working Group studied the architecture and the front-end processing and the other a common approach to the application software needed to run the CERN accelerator complex. The paper presents the technical conclusions of their work and the policy to implement it, taking into account the necessity to operate both machines without interruption of the Physics Program.

I. INTRODUCTION

The complex of CERN accelerators is divided in two sets of different characteristics. The PS set is constituted of ten different accelerators which represent the source of all particles accelerated in CERN (maintained by PS division); they are small and mainly fast cycling machines. The SL set is composed of two bigger machines, the SPS and LEP which are slow cycling accelerator or particles colliders (maintained by SL division). The two sets are, broadly speaking, separated by the Swiss-French frontier.

The PS and SPS accelerators control systems were conceived and implemented some 15 years ago. They are based on 16 bit computers, with a proprietary operating system and a star network. These components do not permit the use of modern industrial software packages and communication standards. Their maintenance is expensive and is becoming more and more difficult. The consolidation of the control systems has become necessary and urgent, and it was felt that one should profit from this consolidation to aim at a real convergence of CERN's accelerator control systems.

In order to work out a common technical solution, the collaboration between the PS and SL control groups has been reinforced considerably since the beginning of the year 1990. A common consolidation project is the result of this collaboration and it was elaborated by working groups of the two divisions. Joint working groups were set up to study the different aspects of the project and to reach the necessary consensus on what should be done. [1]

The first working group designed the common control system architecture, the front end processing and discussed the network characteristics, the local control facilities and the interface between the controls and equipment groups.

The second working group defined a common approach to the application software needed to run the accelerators, discussed the programming environment and the possible software tools and studied the future layout of the work place in the control rooms.

Other groups, linked with the two previous ones, worked on specific subjects like the Equipment Control Protocols, the common on-line data base, the Man Machine Interface, the Timing and Synchronization problems. They generally worked out common solutions which are today in the implementation phase.

II. ARCHITECTURE

The new Common Control System consists of three layers: Figure 1

- the control room layer with its consoles and central servers;
- the front end computing layer distributed around the accelerators;
- the equipment control layer with the Equipment Control Assembly (ECA) crates which form part of the equipment.

The hardware and software used on each level reflect the considerable variety of accelerator components to be controlled. The new architecture offers more flexibility and will allow continuous partial upgrading as technology evolves.

A. Control Room Layer

It must fulfil two main functions:

- Provide the operators with a reliable, user-friendly interface to the accelerators. Modern workstations running the commercial software package X - windows with a suitable commercial tool kit to construct the user interface were chosen. The operator work places are very demanding in terms of graphic and interaction facilities. The key point in selecting the new platforms is the interoperability with the existing equipment and the portability of the software between them. We hope the choices made, UNIX operating system (OSF based in the future) with X-Window and Motif tool kit and communication by the TCP/IP protocol, will allow a smooth transition from one generation of hardware to another.
- Offer a number of central services through servers (which are generally more powerful machines of the same family as the workstations). These central services can be the coordination of various control tasks, central data and file storage, model computing and collection of alarms.

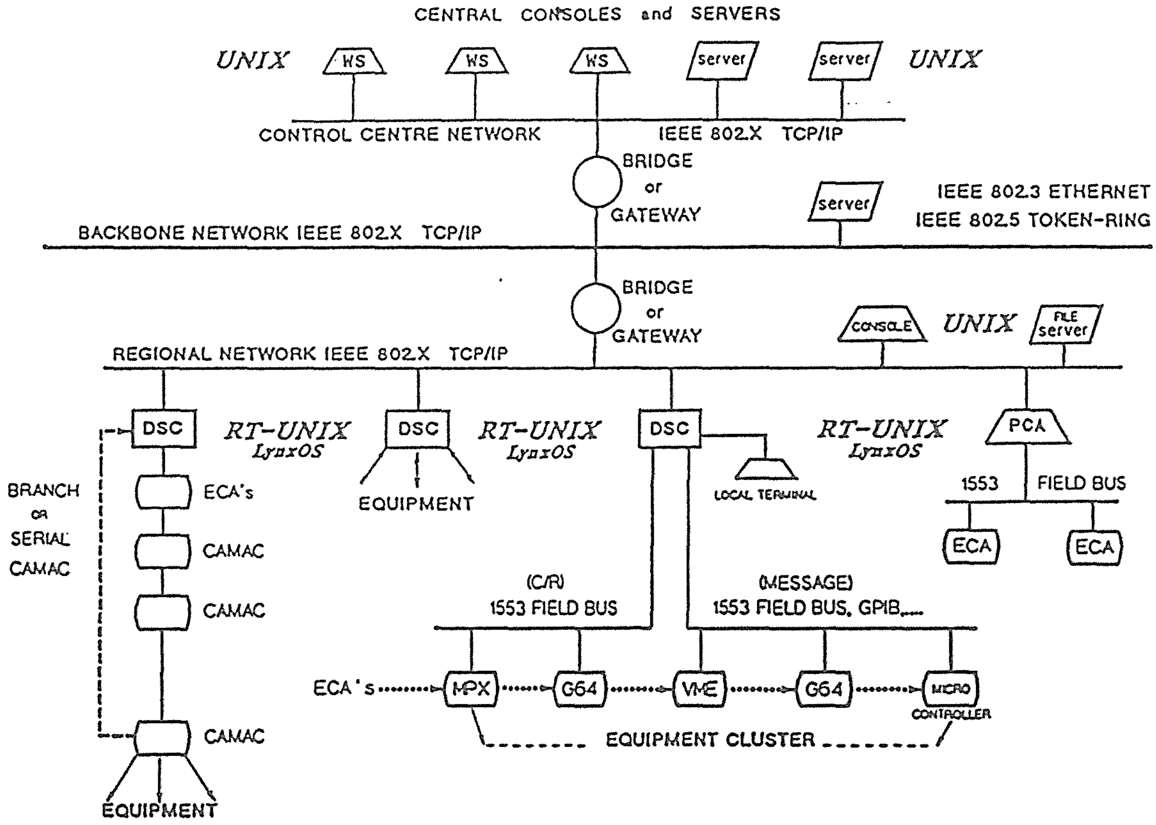


Figure 1. Control System Architecture

B. Networking

The communication between the Control Room layer and the Front End Computing layer, as well as the communication within these layers, is based on modern standards using TCP/IP as the main inter networking protocol suite. The network is composed either of Token Ring or Ethernet segments linked together through bridges or routers. However, where required, routers are implemented in order to filter the access to the control systems from offices, laboratories or the outside world.

The future use of FDDI as an additional fast network is in test between Meyrin and Prevessin sites (utilization by both control systems of an ORACLE on-line data base server).

A Remote Procedure Call (RPC) mechanism offers a way for the programmers to call the libraries located in remote computers, hiding as much as possible the network (CERN designed RPC including an interface to the old control network). TELNET and FTP are also available in connection with the TCP/IP protocol suite.

C. The Front End Computing layer:
The Device Stub Controller (DSC)

The Front End Computing layer is centered on the Device Stub Controllers (DSC) which are based on both standards PC and VME crates with 32 bit embedded microprocessors. A diskless Real Time, UNIX compatible, and POSIX compliant, operating system is run in the DSC. The diskless solution was chosen because the disks of the Process Control Assemblies (PCA) of the LEP control system proved to be a weak point and because the back-up procedures and management of files and data are supposed to be easier if storage is less distributed.

The main functions of the DCS are:

- to provide a uniform interface to the equipment as seen from the workstations;
- to provide direct control and acquisition for equipment like beam instruments, interfaced directly to the DSC;
- to act as a master and data concentrator for distributed equipment, interfaced via a field bus.

The choice of the standards for the DSC, PC and VME bus, was not easy. With the limited resources (staff numbers decrease while the number of new projects increases) the Control groups initially intended to provide support for the VME bus only. However, after long discussions, because of the 65 existing LEP PCA (Process Control Assembly) based on PC and because of quite complementary advantages, it was decided to give support to the two standards, PC and VME bus. This decision mainly became possible with the availability of an open Real-Time, UNIX compatible, operating system which runs on both platforms. Figures 2 and 3

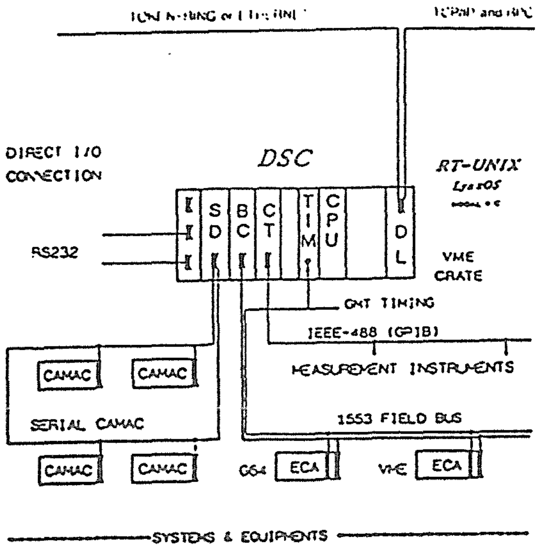


Figure 2. Device Stub Controller

This Real Time UNIX compatible operating system, LynxOS, has been chosen to allow running several tasks concurrently (beam measurement, statistics, alarms, diagnostic programs), and to provide a fast and deterministic response to external events when necessary (Pulse to Pulse Modulation for example). It will make it relatively easy to port the XENIX system libraries and servers developed for the LEP PCA to the new operating system.

To provide disk service for the diskless DSC and workstations (as well as boot server, secure disk space for application and data and back up facilities) file servers will be made accessible. Central file servers are used in general. However, in order to keep the load on the backbone network acceptable and to maintain local control alive in case of communication problems, it has been decided to implement regional file servers on important network segments.

Associated to the DSC, one can distinguish between three different types of local man machine interface:

- Local display (for the DSC based on VME): a graphic VME module is directly driven by the CPU. It could be completed by a touch-panel when a more convivial interface is needed for local operation.
- Local terminal: connected to the CPU of the VME crate with an RS232C line. This access can be used during debugging or testing.
- Regional console: a workstation, a PC with UNIX or a X-terminal, directly connected to the network segment, could be seen as a local use of the services available in the Control Rooms, and can be directly used as a local terminal with TELNET connection.

D. Equipment Control layer

The control crates of the third layer, the Equipment Control Assemblies (ECA), are connected to the DSC via field buses. Since no predominant standard exists, a certain variety of solutions, both for the ECA and for the field buses, is considered to be acceptable. As the main front end computing device is supposed to be the DSC, the computing on the ECA level should be reduced to a minimum.

E. Field buses

In the present CERN accelerator Control Systems, three field buses are used to a large extent: the 1553 field bus [2] and the GPIB in the LEP area and serial CAMAC in the PS area and the SPS experimental areas. At the PS complex, the CAMAC crates will be controlled directly by a VME module that drives the serial loop [3] whilst in the SPS experimental areas the CAMAC crates will be connected to the VME or PC DSC via the VICbus. The SPS main ring equipments will mostly be interfaced to the 1553 field bus. Proprietary field buses delivered with turn-key industrial systems will be interfaced directly to the DSC. [4] Due to the large investment in the associated interface equipment, all three field buses will be supported in the DSC environment.

F. Support to Equipment groups

The equipment groups may need to develop software to control their equipment either in the DSC itself or at the ECA level. When the DSC is directly dedicated to an equipment through interface cards, the control group will offer assistance (cookbook, basic driver "frame") to write the drivers accessing the specific hardware. When the equipment is connected to the DSC through a standard field bus , the Control Group will

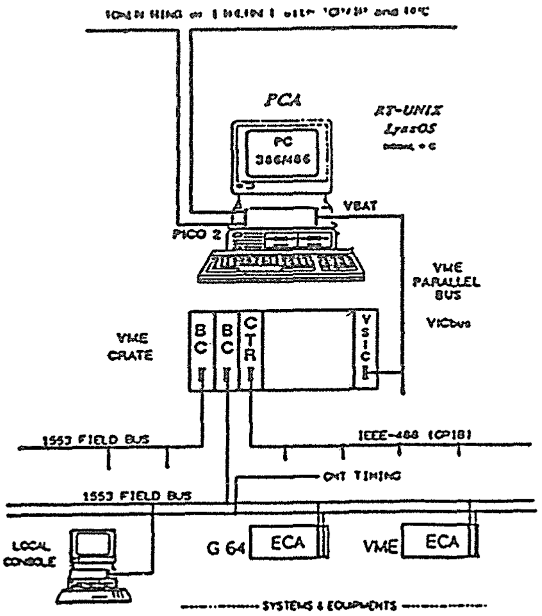


Figure 3. Process Control Assembly

provide the equipment access functions in the DSC. At the level of the ECA, the use of OS9 and LynxOS will get some support from the Control Groups.

G. Proposed Equipment Control Protocols [5]

The best result for the control of a given equipment is obtained when the Equipment Group is totally responsible for its implementation. It is proposed, and already in the beginning of implementation, to use a general operational control protocol which leaves the equipment specialists dealing with all the intricacies of their equipment (either at the level of the DSC or the ECA, where applicable), whilst the control groups take care of the operational requests for this equipment as well as the communication part.

In this scenario, the accelerator operation crew defines a uniform frame for a given equipment interaction, the specialist realizes the specific, local software in the most adequate way for each device, and the controls provide a uniform software interface to translate the operator requests into control sequences for the equipment.

III. APPLICATIONS

A. Basic buildings blocks and tools

The basic building blocks and the tools for the Applications are centered around Open Software Standards. UNIX is confirmed as the operating system of the workstations, the development environment of the DSC and the network servers. The DSC operating system is a UNIX compatible system with real-time performance. (LynxOS) TCP/IP protocol suite and NFS are the basic components for all distributed facilities. Compliance with standards is enforced. The POSIX IEEE 1003.1 interface definition must be respected. The X-Windows is selected as the base graphic system for workstations : for general data presentation facilities and user interactions, all future developments will be based on the X-Windows protocol and the OSF Motif toolkit.

B. Data Management

A well supported set of data management facilities inside the CERN accelerator control systems is needed to cope with the large quantity of data provided by modern electronic equipment and beam instrumentation. The study of the database services sub-group concluded that a dedicated PS/SL database server running ORACLE would be useful. The host machine for this service is installed and first tests and measurements concerning the performance of ORACLE as an on-line database are in progress.

C. Software development facilities

The use of modern CASE tools are encouraged for analysis and design of software projects, but one cannot expect to enforce a systematic application for all projects and to all participants. Source code management tools provided with UNIX are available for keeping track of the history of consecutive versions. "C" is the common base language, as it is well integrated in UNIX platforms and offers a good portability (as accepted by the Portable C compiler). The NODAL interpreter (written in C) is available on the DSC and on both workstation platforms in use, DEC and Hewlett-Packard. Fortran may be used, but only for mathematical applications. (mainly modeling)

D. Environment of application software

The development environment should be as similar as possible to the operational environment in order to ease the transition from development to operation and to allow good productivity and maintainability by the knowledge of a single environment. Standard procedures must be established for validation by the users (operation, accelerator physicists, equipment groups) and controls exploitation staff. A complete application program must be designed with integration into the Console Manager in mind; the Console Manager is the supervisor of all activities induced by the workstation.

E. Application run-time environment

No synchronization with the cycle of accelerators will be done through workstations. The equipment cannot rely on software for its protection, and the equipment groups will have to provide means to protect their equipment against unallowed actions through the control system.

F. Man Machine Interface

The major goal is to reach as much uniformity as possible for the operation environment. To preserve the future software investment in MMI, the OSF tool kit MOTIF will be used on top of the well established and widely used X-Windows standard display protocol. The console manager activates and coordinates the various processes according to the requests of the user. Synoptics will be widely used in the controls of the accelerators (as they are also the natural tool offered for various industrial equipment); the DVDraw/DVtools product, encapsulated in a man machine interface, will be used for the production of such synoptics. [6] [7]

G. General Control room environment and operator desk

The operation of accelerator clusters through the control system is done by operator teams working 24 h a day in the central control rooms and using consoles as the main tool for machine interaction. The notion of "work place" is now preferred to the one of console. The powerful workstations are the main interactive tool composing a work place; they are the tools to select, visualize and drive the batch of application programs to control a selected part of an accelerator. The work place is completed by tools to observe and select analog and video signals, and to display alarms.

Beside the workplaces, an accelerator control center must contain tools to display general information summarizing the status of the accelerator complex, means to communicate with local building and other control centers, means to access other network facilities, and tools to manage personal safety which must be treated separately from accelerator control.

IV. CONVERGENCE POLICY

Because of the large hardware and software investment needed, the limited manpower available today, the similarity of the control requirements for the various machines and the increasing complexity of the software tools for running them, it is mandatory to reduce the diversity of control implementations at CERN.

The constraints imposed on the convergence by the history of the different machines, the large amount of investment in existing hardware and software, the different types of machines to be controlled, and the habits adopted by the operation crews interacting with these machines must be taken into account when defining a common PS/SL policy.

This common policy will be based on a common control system architecture, using well established industrial standards for control networks, communication protocols, equipment, operating systems and man machine interface. This architecture will profit from and enforce the use of Open-System products supported by many manufacturers and consortiums. (UNIX, OSF, X-Windows for the software; Ethernet and Token-Ring with TCP/IP for the control network; UNIX workstations for the man machine interface; PC and VME for the DSC and equipment interfaces).

As much software as possible must be written independently of the development platform (using industrial products), opening the possibility to transfer application programs between the different accelerators.

An other main goal of the common policy must be to preserve the existing hardware and software investment. The application software developed for the exploitation of the accelerators represents hundreds of man-years of investment which has to be maintained, improved and optimized to ensure uninterrupted operation. The proposed architecture makes provision to integrate the hardware investment in CAMAC and MPX in providing adequate tools to access these crates. The software investment in the SL-PCA is preserved by using a UNIX compatible real time operating system in the DSC.

V. CONCLUSION

This is the summary of the consensus which has emerged from the discussions about the architecture of CERN's accelerator control systems, their main components and the general aspects of the application software. A major concern was to preserve the existing hardware and software investment, together with a non-duplication of the effort inside CERN's accelerator community.

Our main aim was to propose an architecture which permits continuous partial upgrading, as technology evolves, in order to avoid any "big bang" operation in the future. The main components of our control systems are based on open standards, for the hardware as well as for the software, to become independent of a given manufacturer.

Finally, to monitor the progress of the consolidation project and to ensure that all our efforts stay directed towards a common goal, the working groups (or at least an emanation of the two working groups) continue to meet during the implementation of the different steps of the project.

VI. REFERENCES

[1] The PS and SL controls groups, "PS/SL Controls Consolidation Project", Technical Report, CERN PS/91-09 (CO), CERN SL/91-12 (CO), April 1991.

[2] R. Rausch, "Control of the large European LEP and SPS Accelerators Based on the 1553 Field Bus", CERN SL/9005 Report, Conference on MIL-STD-1553-Band the Next Generation, London, United Kingdom, 29-30 November, 1989.

[3] W. Heinze, "Driving serial CAMAC systems from VME crates", these Proceedings, November 1991, ICALEPCS, Tsukuba, Japan, 1991.

[4] K.H. Kissler, R. Rausch, "New Control Architecture for the SPS Accelerator at CERN", November 1991, these Proceedings, ICALEPCS, Tsukuba, Japan, 1991.

[5] WOPRO Working Group, reported by G. Benincasa, Control Protocol : "The Proposed new CERN Standard Access Procedure to Accelerator Equipment : Status Report", these Proceedings, November 1991, ICALEPCS, Tsukuba, Japan, 1991.

[6] M. Boutheon, F. Di Maio, A. Pace, "General Man-Machine Interface used in Accelerators Controls: some applications in CERN/PS Control System Rejuvenation", these Proceedings, November 1991, ICALEPCS, Tsukuba, Japan, 1991.

[7] P. Antonsanti, M. Arruat, J.M. Bouché, L. Cons, Y. Deloose, F. Di Maio, "Workstations as consoles in the CERN/PS complex: setting up the environment", these Proceedings, November 1991, ICALEPCS, Tsukuba, Japan, 1991.

New Control Architecture for the SPS Accelerator at CERN

K.H. Kissler and R. Rausch

European Organization for Nuclear Research

CERN - 1211 Geneva 23 - CH

Abstract

The Control System for the 450 Gev proton accelerator SPS at CERN was conceived and implemented some 18 years ago. The 16 Bit minicomputers with their proprietary operating system and interconnection with a dedicated network do not permit the use of modern workstations, international communication standards and industrial software packages. The upgrading of the system has therefore become necessary.

After a short review of the history and the current state of the SPS control system, the paper describes how CERN's new control architecture, which will be common to all accelerators, will be realized at the SPS. The migration path ensuring a smooth transition to the final system is outlined. Once the SPS upgrade is complete and following some enhancements to the LEP control system, the operator in the SPS/LEP control center will be working in a single uniform control environment.

I. HISTORIC REVIEW

The SPS control system was designed in 1972 and brought into operation in June 1976. By that time no standard communication network protocol existed and pioneering work had to be done to interconnect initially some twenty minicomputers located in 6 equidistant equipment buildings and one central control room around the 7 Km circumference of the SPS accelerator ring. The minicomputers used were NORD10 from Norsk Data with 16 KByte core memory and 128 KByte drum mass storage. The equipment interface consisted of CAMAC crates connected to the computer's I/O bus with CAMAC modules linked directly to some of the beam instrumentation while all other equipment was controlled via a CERN designed multiplex (MPX) system composed of a serial field bus, MPX crates and user dedicated MPX modules.

On the software side, the manufacturer's operating system has been modified to suit the particular real-time control requirements of our distributed multiprocessor environment. While most of the software drivers were written in computer assembly code, an interpreter, called NODAL, has been developed to provide easy interaction between the operator and the equipment connected to CAMAC, remote access facilities and network functions. Every computer had a resident NODAL interpreter allowing to use it in interactive mode and in stand alone operation for test and commissioning purposes. No one computer would be the over-all master of the control system and the message transfer system was designed so that any computer could pass a message to any other without a preset master-slave relationship, implying that the system was completely symmetrical and transparent [1].

II. PRESENT SITUATION

Since the start-up of the SPS in 1976, the control system has been extended to cope with the changing requirements of the accelerator which was initially designed as a pulsed proton accelerator for fixed target experiments, then modified to act as a proton/antiproton storage ring for collider physics, and now also as an injector to LEP, accelerating electrons and positrons interleaved with proton acceleration. Such evolution has required a great flexibility of the control system with the ability to modify programs as necessary in a simple way and to add computers, network links and equipment interfaces where and when required, sometimes even during the exploitation of the accelerator complex.

Today the SPS control system is composed of 52 operational process and central computers (NORD100) interconnected by a multi-star network with 6 Message Handling Computers (MHC). The interfacing between computers and accelerator equipment is done by CAMAC (72 crates and 450 modules) and by the MPX system (693 crates and 5500 modules). Figure 1.

With the need to operate the SPS in a supercycle mode when using it as a LEP injector, major additions had to be made to the control system since about 1986. A more flexible and versatile exploitation of many accelerator components like beam monitors and pulsed power converters was required. At the equipment level, this requirement has led to the use of 8 Bit and 16 Bit microprocessor based systems, embedded in G64bus and VMEbus crates. These Equipment Control Assemblies (ECAs) are connected to the appropriate NORD100 process computer via 1553 field bus segments and a VMEbus crate housing the bus controllers and linked to the computer's I/O bus.

In the accelerator control room, Apollo workstations where installed to cope with the more complex supercycle operation. These communicate with each other and with an

Apollo file server via an IBM Token-Ring network. A gateway from the Token-Ring to the old proprietary star network of the SPS permits the Apollos to communicate with the NORD100 computers.

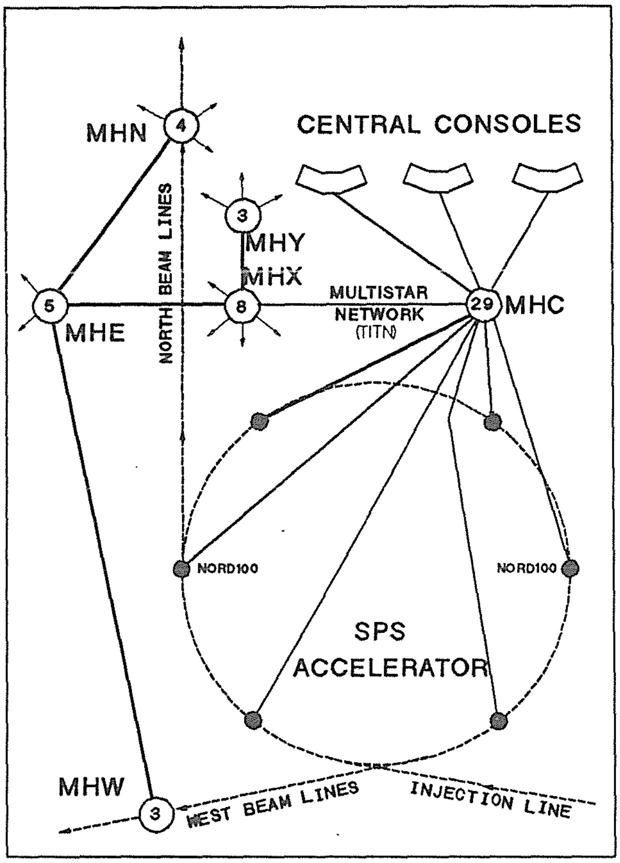


Figure 1. Present SPS Control Network.

The SPS was also used as a test bed for the LEP control system [2]. To this end a full network infrastructure including a backbone Token-Ring around the accelerator and a Time Division Multiplex (TDM) system for long distance transmission were installed at the SPS well before this was possible in the LEP tunnel. All SPS auxiliary buildings were equipped with LEP type Process Control Assemblies (PCAs). Each of these consists of a PC linked to a VMEbus crate via a 1553 connection, the VMEbus crate containing the bus controllers for the 1553 field buses which extend to the equipment. This infrastructure has permitted to validate the technical choices for LEP, but will now also serve in the framework of the new SPS controls.

III. CERN's NEW CONTROL ARCHITECTURE

The new control system architecture on which the PS and the SPS/LEP control groups have agreed after extensive studies and consultation with the control system users, is laid down in a comprehensive report [3] and presented in some detail in another paper at this conference [4]. Here only a short summary will be given to make this paper self-contained.

A. Architecture

The new architecture is based on three computing layers:

the *Control Room Layer* which provides the operator with a user friendly interface using modern UNIX workstations with OSF Motif and X-Windows graphic software packages. This layer comprises also a number of central servers for data and file storage, model computing, alarm collection, network management and for an on-line relational database.

the *Front End Computing Layer* with its so-called Device Stub Controllers (DSCs) based on Personal Computers (PCs) and VMEbus systems, running a POSIX compliant real-time operating system. Secure disk space is provided locally or centrally by dedicated file servers. Local access and graphics are standard features of PCs and are provided on VMEbus crates by terminals and displays connected to dedicated modules. Alternatively, a local terminal server or an X-Terminal, connected to the Ethernet segment, will provide access to all computers (local or remote) on the control network.

the *Equipment Control Layer* with Equipment Control Assemblies (ECAs) connected to front end computers via field buses and to the equipment by mean of general purpose or dedicated electronic modules. Branch or serial CAMAC and 1553 field bus will allow to integrate existing CAMAC and MPX crates at this level. Figure 2.

B. Network and Communication

The data communication between CERN's different Main Control Rooms will be done via a Fiber Distributed Data Interface (FDDI) network. The FDDI is a Local Area Network (LAN) protocol defined by the ANSI X3T9 Standards Committee. The network has a ring topology, uses token passing access, a fiber optic transmission media, provides a 100 Mbit/s data rate, can span distances up to 2000 meters and supports up to 500 network nodes. This FDDI network will allow to share common PS-SPS servers and an ORACLE data base computer.

The communication within and between the Control Room and the Front End Computing Layers is based on modern LANs: Ethernet and Token-Ring conforming to IEEE 802.X International Standards and using the TCP/IP protocol. Program communication will rely on a CERN designed Remote Procedure Call (RPC) until a standard is defined and industrial products become available. Where long distances are involved (>500 meters) a Token-Ring backbone is implemented with transmission over TDM equipment conforming to the CCITT G700 Standard.

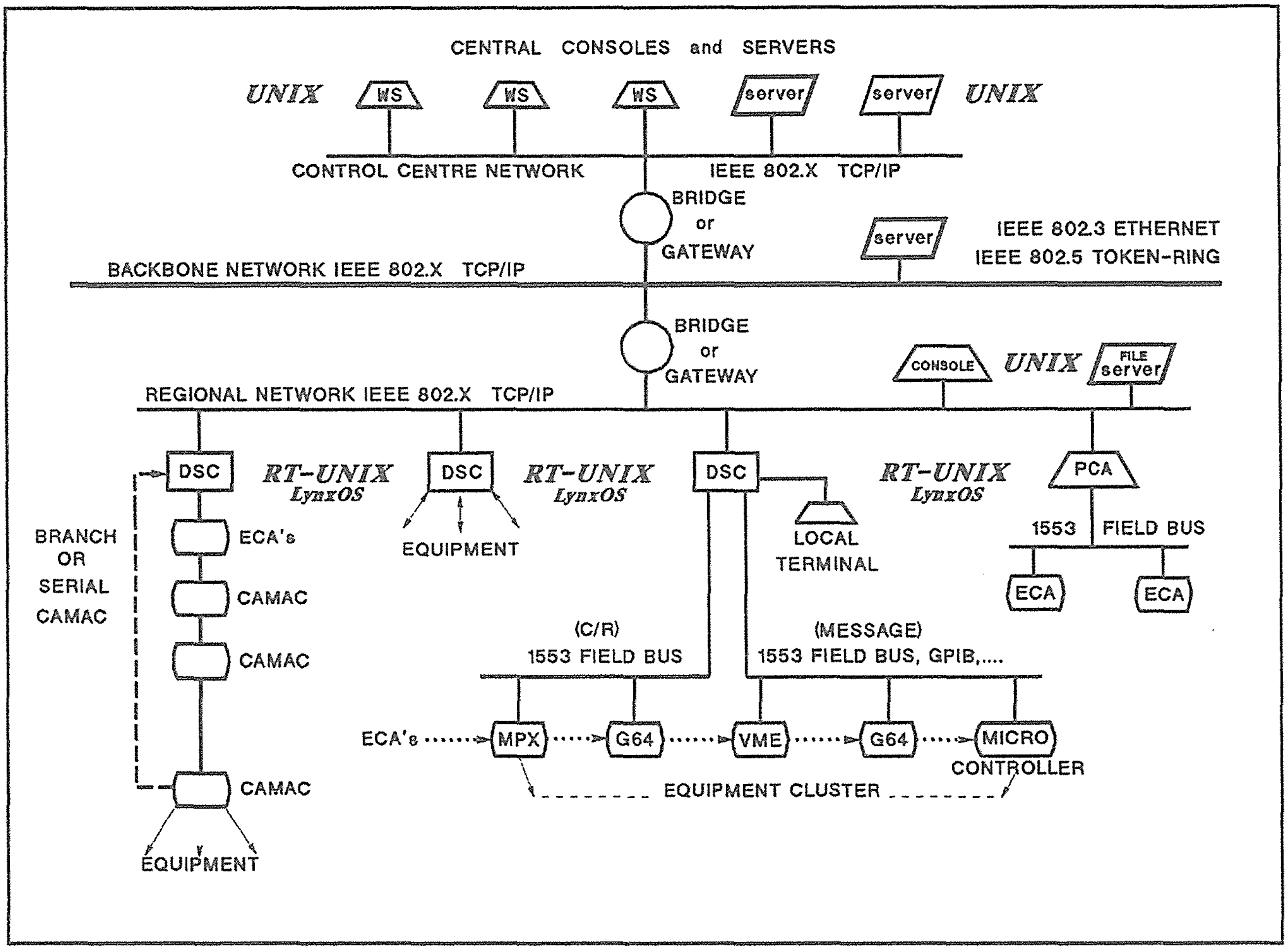


Figure 2. New Control Architecture.

C. Operating Systems

It has been decided to use a UNIX environment for the control of CERN's accelerators to the largest possible extent.

UNIX is the only operating system supported by virtually all computer manufacturers, from the PC clones up to the large CRAY or IBM mainframes. UNIX provides *scalability* which allows to move programs and data from smaller machines to larger ones. UNIX avoids getting trapped in a proprietary environment: overcharges, poor services, failure to provide an upgrade path, company which goes bankrupt, etc.. In this case the *portability* of UNIX permits to move programs and data to another brand of hardware. For the programmers it is easier to port an application between any two versions of UNIX than any two closed operating systems. In addition, combining UNIX with a public networking standard (TCP/IP), it provides *interoperability*; the ability of networked computers to share files and applications.

For the front end process computers, it has been decided to use a commercial real-time operating system, which runs on both industry standard 386/486 PC/AT compatibles and VMEbus CPU modules, based on the Motorola 68030 microprocessor, and which complies with the IEEE 1003-(1990) POSIX standard (POSIX.1 for the application program interface, POSIX.2 draft for shell and utilities and POSIX.4 draft for the real-time extensions).

The acronym POSIX stands for Portable Operating System Interface. POSIX is written by IEEE working groups as a US standard and acquires international status via its acceptance by ISO/IEC (International Standard Organization International Electrotechnical Commission) as International Standard 9945, [5].

POSIX is a software interface standard which guarantees portability of application source code but is not an operating system implementation standard. Real-time POSIX addresses the full range of real-time systems, from full scale UNIX down to small embedded kernels with the highest demands on true real-time performance.

It is well known that basic UNIX is not designed to be a real-time system. Its goal is to give a fair share of system resources to every user, not to have, for instance, a high priority process taking the CPU and keeping it as long as necessary. On the contrary, real-time POSIX ensures an operating system the ability to provide a required level of service in a bounded response time.

Following a CERN tendering procedure LynxOS has been selected.

IV. IMPLEMENTATION OF THE NEW ARCHITECTURE AT THE SPS

A. General consideration

CERN's new control system architecture described above defines the general framework and the guide lines to be respected when replacing the old SPS system. It leaves, however, the freedom to take account of other facts proper to the SL division: the existence of and experience with the LEP control system as well as the infrastructure created during the preparation of the SPS as a LEP injector. Both facts, in particular the experience with the LEP system and the desire to create a uniform controls environment for the two accelerators, have had and are still having a strong influence on the detailed choices made for the SPS.

B. The Control Room Layer

SPS and LEP are both operated from the same control room. Apollo workstations running under Domain OS 10.3 are currently used to provide the operator interface to LEP and also to control the supercycle operation of the SPS. More powerful machines of the same workstation family offer central services, such as data and file storage or model computing. Recently, Hewlett-Packard workstations, model 9000/400 and 9000/425, still running under Domain OS, have been added to the system.

We are now in the process of introducing a few DEC 5000/200 workstations, running ULTRIX 4.2, into the SL control system. Their main function is to provide centralized secure disk space for applications and data, to provide disk service for diskless front end processors and to be the bootstrap servers for front end processors, workstations and X-terminals. The DEC stations were mainly chosen in the framework of the collaboration with the PS division, permitting to set up common services like a common ORACLE database.

In the near future, HP 9000/730 and 9000/750 workstations will make their appearance at SL. Running HP UX 8.05, they will add to the diversity of UNIX flavors used at the level of the SL control room layer. This diversity is considered to be temporarily acceptable and should largely disappear with the future introduction of the OSF 1 operating system for both DEC and Hewlett-Packard workstation families.

While the general data presentation and user interaction in the SPS/LEP control room are presently still based on the Apollo proprietary user interface management system Domain/Dialogue, all future developments will make use of the X-windows protocol and OSF Motif (toolkit, User Interface Language, resource manager, style guide).

C. The SPS control network

As mentioned before, a backbone Token-Ring was installed around the SPS a few years ago already, completed by local Token-Rings in the main control room and in the SPS auxiliary buildings. To this installation, local Ethernet segments will now be added in order to guarantee a better connectivity of equipment from different vendors to the network.

D. Front End Computers

All NORD100 computers will be replaced progressively by front end computers based on PCs or on VMEbus crates depending on the type of equipment they control, on the necessity to have local mass storage, graphics facilities and local interactivity for testing purposes. It is essentially the responsibility of the Equipment Group in charge to decide which technical solution is best.

Typically, equipment like beam instrumentation which needs fast response and high throughput, will be directly connected to modules located in VMEbus front end computers. These in turn will be connected to the local Ethernet segment and share either a local or a central file server. Such an arrangement has recently proved to be very successful at LEP where some 40 VMEbus crates with direct connection to the network are used for closed orbit correction [7].

For systems with large numbers of identical Equipment Control Crates (ECAs), PC based front end processors will be used to regroup the ECAs via a 1553 field bus and enable local supervision, alarm reduction and alarm identification before transmission to the main control room. The Process Control Assemblies (PCAs) described in chapter II will be the most frequent configuration used in this context.

Local consoles, PCAs and VMEbus based front end computers installed in the equipment buildings will operate under LynxOS and will be able to run the same programs as the central workstations and servers located in the main control room, all will communicate over the control network using the TCP/IP protocol.

E. Equipment Field Buses

In LEP, ECAs usually contain microprocessor units and are addressed in message mode via the 1553 field bus.

This mode of operation will also be used in the SPS whenever old obsolete electronics will be replaced by new modern boards housed in intelligent control crates [6].

However, many of the existing MPX crates are still working reliably and will be preserved. In this case the old MPX multidrop bus, connecting them via a CAMAC crate to the NORD100 computers, will be replaced by a 1553 field bus, generally controlled by a PCA. As no microprocessor is used in the MPX system, the 1553 field bus will be working in command/response mode and a special 1553/MPX interface card has been developed for this purpose.

It is at the field bus level that the General Machine Timing (GMT) is distributed to the equipment. For economical reasons and cabling simplicity, both control and timing signals are transmitted over two twisted pairs of wires in the same multidrop cable [8].

Other field buses can be used at the SPS to link ECAs to the front end processors. Industrial systems are often delivered with a proprietary field bus: Bitbus, Fip, Filbus, J-bus, Profibus, Proway, etc.. To be integrated into the SPS control system these industrial systems must be delivered with a VMEbus or a PC/AT bus controller and an adequate software driver, compatible with the LynxOS real-time kernel, must be available. In addition, the manufacturer's equipment control protocol must be known and be converted to the CERN standard to allow homogeneous access to all equipment of the accelerator from the operator's workstation.

The experimental areas of the SPS present a special case where the existing field bus, a serial CAMAC loop, will be replaced by Ethernet. A PC connected to the local Ethernet segment will be installed in each of the stations which regroup the equipment control hardware. This hardware will continue to be controlled by CAMAC modules and their CAMAC crate will be linked to the PC by the parallel Vme Inter-Crate bus (VICbus).

V. THE MIGRATION PATH

A major step of the migration towards the new control architecture is planned to take place during the 1991-92 winter shutdown of the SPS.

The existing gateway between the old multistar network and the new Ethernet/Token-Ring infrastructure only allows communication in command/response mode from the Apollo workstations to the NORD100 computers. It will be replaced by a new bidirectional gateway which will enable the NORD100s to address to services on the new network. The library computer used for data and file storage and other NORD100 based servers, all operating with obsolete disk units, will then be removed and their tasks taken over by modern workstation based servers.

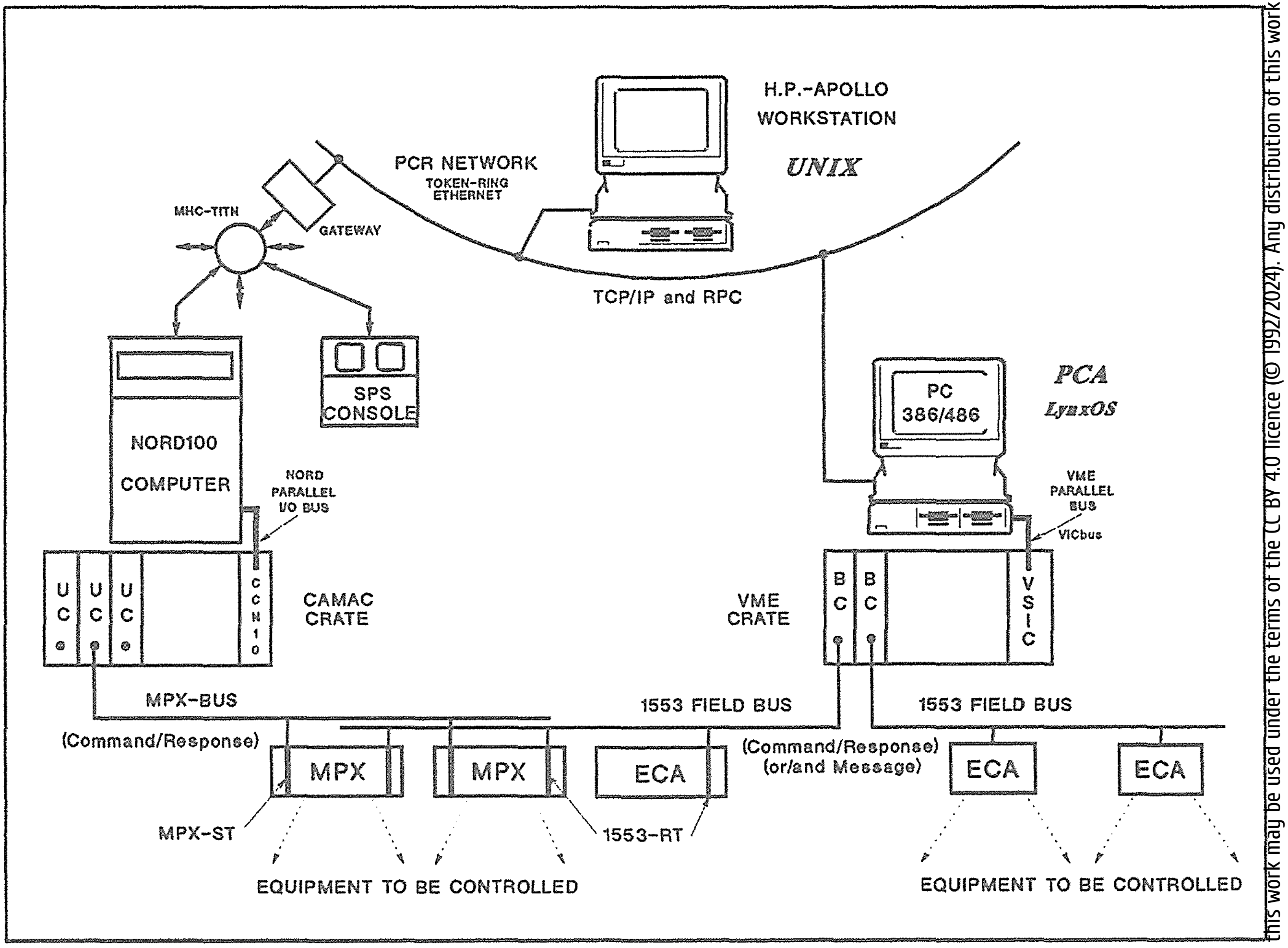


Figure3. Migration from NORD100 to PCAs

The new gateway will be installed during the month of October and put into operation during the last two months of the year. This new gateway will first operate in parallel with the old one to allow for software debugging during accelerator operation while keeping the possibility to switch back at any time in case of problems.

At the same time a new version of the CERN RPC (Remote Procedure Call) will be introduced as well as an improved version of the NODAL interpreter, written in C language, providing additional functionalities.

During this winter shutdown existing VMEbus based front end computers and PCAs will be connected to local Ethernet segments which in turn will be bridged to the Token-Ring backbone and all PCAs will be upgraded by using the VICbus connection to the VMEbus crates, instead of the 1553 link.

It is also foreseen to install LynxOS 2.0 in the PCAs, in replacement of the present SCO XENIX.

This upgrade combined with the general use of NFS (Network File System) and the hardware improvements described above is expected to provide a better overall performance and response time compared to that of the present LEP control system.

In the main control room DEC file servers have already been installed to provide secure file space for programs, applications, data storage and remote boot facility for the PCAs and the VMEbus based front end computers.

Existing Hewlett-Packard workstations will be used with X-windows and Motif to write new application programs for the systems linked to the new infrastructure. Thus the complete chain, from the operator workstation down to the ECAs including the network and the new front end computers can be tested and debugged.

To facilitate the migration of application programs from the old NORD100 consoles to the Hewlett-Packard workstations, for equipment remaining to be controlled by MPX crates, an alternative access will be possible for test and debugging purposes during the transition phase. Figure 3.

VI. CONCLUSIONS

Modernizing the control system of a running accelerator is a difficult task. Since its first operation in 1976 the SPS has undergone a number of smooth upgrades to cope with the changing demands of the accelerator exploitation. Generally more installation of the same kind of equipment or replacement of mini computers by more powerful ones of the same family, was fairly straightforward.

This time, a drastic change is required. Computers, networks, local intelligence, application and system programming have changed totally over the last 15 years. Hardware and software standards are available but the job has not become easier for the control specialists. New skills are required and system integration is the buzz word.

To preserve the integrity of the accelerator operation, the migration work must be done in well defined steps. The test bed chosen for the validation of the new architecture and the methodology adopted must be representative and include all aspects of the project. The overall planning is essentially determined by the duration of the annual accelerator shutdown and available manpower, particularly in the field of application software.

VII. ACKNOWLEDGEMENTS

Many members of the PS and SPS/LEP control groups have contributed to the definition of CERN's new control system architecture and to the application of the ideas within the SPS environment. The authors wish to thank them all. Particular thanks are due to J. Altaber and F. Perriollat who where co-authors of the first proposal to harmonize the controls of all accelerators at CERN.

VIII. REFERENCES

[1] M.C. Crowley-Milling, "The Design of the Control System for the SPS," CERN 75-20 Report, 22 December, 1975.

[2] P.G. Innocenti, "The LEP Control System," International Conference on Accelerator and Large Experimental Physics Control Systems, Vancouver, BC, Canada, 30 October - 3 November, 1989.

[3] The PS and SL Control Groups, (Editors: R. Rausch and C. Serre), "PS/SL Controls Consolidation Project," Technical Report, PS/91-09 or SL/91-12, CERN, Geneva, Switzerland, April, 1991.

[4] R. Rausch, C. Serre, "Common Control System for the CERN PS and SPS," International Conference on Accelerator and Large Experimental Physics Control Systems, Tsukuba, Japan, 11-15 November, 1991.

[5] C. Eck, "Standardization of Real-Time Software POSIX 1003.4," Real-Time 91 Conference, Jülich, Germany, 24-28 June, 1991.

[6] R. Rausch, "Control of the Large European LEP and SPS Accelerators Based on the 1553 Field Bus," CERN SL/90-05 Report, Conference on MIL-STD-1553-B and the Next Generation, London, United Kingdom, 29-30 November, 1989.

[7] G. Morpurgo, "The Software for the CERN LEP Beam Orbit Measurement System," International Conference on Accelerator and Large Experimental Physics Control Systems, Tsukuba, Japan, 11-15 November, 1991.

[8] C.G. Beetham, K. Kohler, C.R.C.B. Parker, J.B. Ribes, "The Transmission of Accelerator Timing Information around CERN," International Conference on Accelerator and Large Experimental Physics Control Systems, Tsukuba, Japan, 11-15 November, 1991.

The Next Generation Control System of GANIL

T.T. Luong, L. David, E. Lécorché, M. Ulrich
Grand Accélérateur National d'Ions Lourds
BP 5027 - F 14021 CAEN Cedex, FRANCE

Abstract

The existing computer control system of GANIL is being renewed to fulfil the increasing requirements of the accelerator operation. This medium term major improvement is aiming at providing the physicists with a wider range of ion beams of higher quality under more flexible and reliable conditions.

This paper gives a short description of the new control system envisioned. It consists of a three layer distributed architecture federating a VAX6000-410/VMS host computer, a real time control system made up of a dual host VAX3800 and workstation based operator consoles, and at the frontend segment: VME and CAMAC processors running under the VAXELN operating system, and programmable logic controllers for local controls.

The basic issues with regard to architecture, human interface, information management, ... are discussed. Lastly, first implementations and operation results are presented.

I. INTRODUCTION

The GANIL laboratory has been operating since 1983 an accelerator complex consisting of three machines in cascade : a compact injector cyclotron and two fourfold separated sector cyclotrons (Fig. 1).

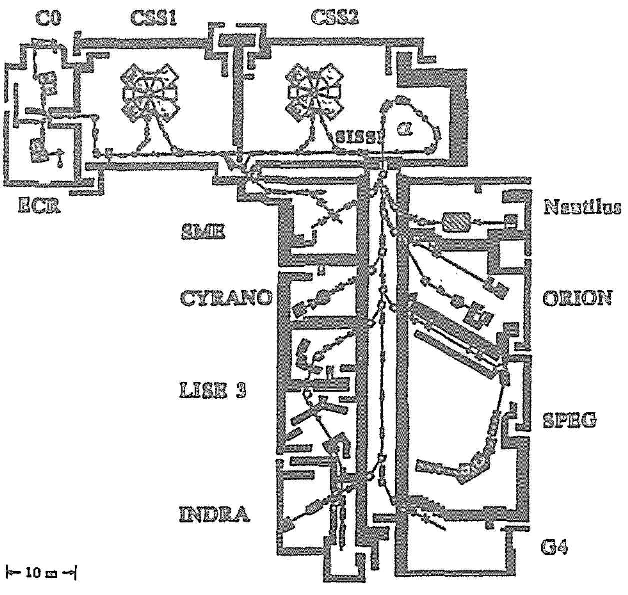


Figure 1. Accelerator and Experimental Areas

This facility provides the experimenters with fast heavy ion beams for fundamental research in the fields of nuclear physics, atomic physics and solid state physics, as well as for industrial applications.

Significant upgrades were carried out these last few years to augment the energy of heaviest ion beams and to increase their intensities by making use of new ECR source. Acceleration at GANIL henceforth encompasses ion species, from carbon to uranium, with beam energy ranging from up to 95 MeV per nucleon for the ions with masses up to 40 u to 24 MeV per nucleon for the heaviest ions.

The rejuvenation of the GANIL computer control system is under way, aiming at two main goals : 1/supersedes the present control system which is technologically outmoded and driven to its ultimate capabilities, 2/matches the performances of the emerging control system with the widening scope of the services in a large variety of domains (beam setting and tuning, surveillance, diagnostics, expertise,...) within an operator friendly environment.

This paper emphasizes the main topics to be considered when designing and implementing our next generation control system. In particular, stress is laid on using : 1/acknowleged industry or international open standard hardware and software products to achieve minimization of investment over the life of the system, 2/modular structures to make easier future expansions.

II. ACCELERATOR CONTROL SYSTEM

II.1. General Layout

The first generation control system adopted a centralized architecture built with a 16bit minicomputer (MITRA 625) which ruled over other kinds of processors devoted to local or ancillary tasks : 8bit (JCAM10/INTEL 8080) and 16bit (DIVA/MC68K) microprocessor CAMAC controllers, programmable logic controllers (APS30-12 and PB400). These processors are connected to the MITRA via two bit-serial 2.5 MHz CAMAC loops which bind up 40 crates with about 800 attached modules.

This tight coupling with the computer MITRA considering architecture and non portable software makes the control system vulnerable with regard to collapse, obsolescence and ageing of that computer.

In contrast, the GANIL control system to come is based on a distributed architecture. Intelligence is therefore handed over to local processors which are responsible for dedicated field operations. The chosen topology features three functional levels which intercommunicate by means of an Ethernet local area network (LAN) :

1. The HOST level provides a general purpose and cosy environment for software programming, debugging, off-line calculations and displays, database management, and simulation. It is built around 1/ - a midrange DIGITAL EQUIPMENT computer, the VAX600-410, connected to video terminals through DECservers, 2/ - workstations VS3100-76 equipped with 19' high definition color screens for graphics oriented developments.

This level also realizes links with other facilities of the laboratory : compatible PC's and Mac serviced by a NOVELL server, CAD stations and the Physics acquisition Vax-cluster ; in addition, it provides connection to remote physics laboratories via wide area netwoks (WAN).

2. The REALTIME CONTROL level allows operators to control the accelerator by means of appropriate human interfaces. This level is based on a microVax3800, seven workstations VS3100 and X-terminals VT1300 to benefit from the enhanced graphic capabilities within the X11 standard multiwindowing environment. VAXELN controlled VME boards are used to drive the shaft knobs. Man-machine interaction will be emphasized in § II.4. The μVAX3800, which plays a key role in the real time control, is actually a dual host cluster equiped with redundant disks to achieve some kind of "failure tolerance ".

3. The EQUIPMENT level perfoms low level controls with different kinds of front end processors :

- CAMAC controllers (KSI3968 from KINETIC SYSTEMS) and VME controllers (VME300 from AEON, ...) running real-time applications under the VAXELN operating system. These controllers which are referred to as front end controllers (FEC) integrate the RTVAX300 chip, the CAMAC FEC replacing the present serial loop crate controllers.
- Programmable logic controllers (PLC) : S5-135U from SIEMENS, PB400 from TELEMECANIQUE/APRIL.

CAMAC and VME FEC, as well as the SIEMENS PLC are directly connected to the controls Ethernet LAN. Communication between VAX processors and SIEMENS PLC is achieved by DEC software packages : VSH1 which supports the application-presentation - session layers of the OSI/ISO standard and VOTS which provides services of the two next lower layers. The PB400 PLC are connected to the server node μVAX3800 by means of an asynchronous serial link that supports the master/slave JBUS communication protocol. The very first APS30-12 programmable controllers, which are devoid of LAN connexion capability, are phased out.

The general Ethernet LAN, is linked to the sensitive control Ethernet LAN via a bridge chosen for its filtering capability. Communication protocols which are currently DECnet, TCP/IP, LAT will comply with the OSI standard. Fig 2 displays the layout of the future control system.

II.2 Software considerations

Requirements

It is a matter of fact that software is taking a leading part in modern control system, as compared to hardware, with high added value caused by large human effort (many people involved over a long time to carry out).

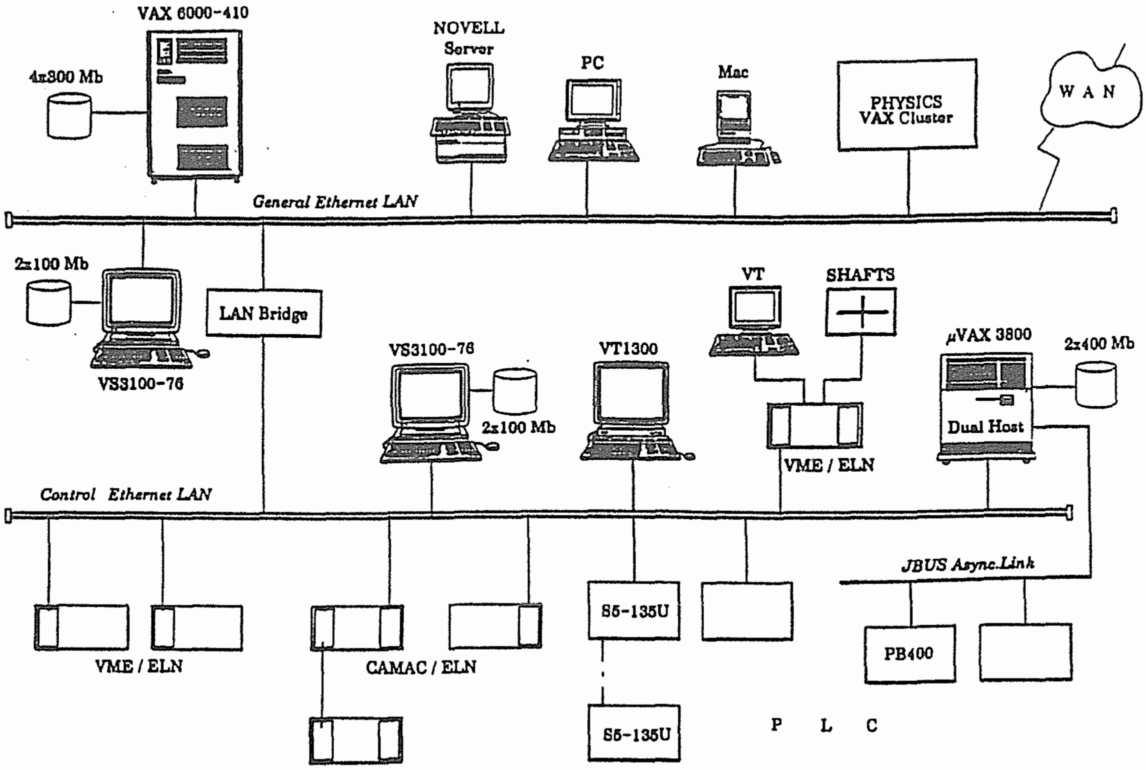


Figure 2. Schematic layout of the future Control System

Our basic requirements are :

- Dependable software to achieve productivity taking into account modern tools and methods.
- Trustworthy and easy operation which implies ergonomics.
- Well guided maintenance.
- Durability. GANIL evolutions, as a physics laboratory may be stressing, unforeseen and of large amplitude. The control system should face such a situation in the smoothest possible fashion.

Choices

To meet our requirements, standardization is the key.

Operating systems : VMS for VAX processors and VAXELN for real time controls are adopted. VAX/VMS is an industry (de facto) standard widely used in physics laboratories, VAXELN is chosen as a mature and powerful product which is tailored for real time performances. It supports a host/target connection over DECnet within a full VAX/VMS compatible environment. Therefore, the host processor VAX6000-410 performs developments and debugs while the targets (CAMAC and VME FEC) execute real-time controls.

Languages : A multilanguage system is chosen with ADA, Fortran and C. These languages comply with AFNOR, ANSI and ISO standards. ADA is our major programming language, while assembly language is relinquished.

Industry softwares : In addition to the software packages devoted to the VMS environment, LAN communication and management, some important software packages are selected :

. the INGRES family from ASK/INGRES devoted to relational database management (more in §II.3)

. the IMAGIN family from SFERCA devoted to supervision (more in §II.4)

. Xwindows and MOTIF (X11R4).

Implementations

A basic software layer has to be designed to meet the specific control requirements of the GANIL accelerator. This so called GANICIEL layer is mainly transparent to the users and is built upon the industry softwares.

It makes widely use of the client-server model and takes into consideration the distributed architecture which allows the clients and the server to run on different processors located anywhere in the LAN.

The following emphasizes the distribution of the GANICIEL functions over the control system levels :

At real time control level

The µVAX3800 assumes the function of a global server that is fanned out into dedicated functional programs to handle remote incoming requests.

Important ones are :

- Initiation of appropriate functions (e.g. communication, tasking, ...) on boot or on resumption.
- Surveillance of networking operation and FEC execution.
- Alarm handling which deals with concentration, storage and display for operator decision, in instant mode or differed mode with customized presentation.
- Data base management

. SQL translation of request stemming from FEC

. Dispatch dedicated run time database to the proper processor

. Update in partial mode for some pieces of equipment or in global mode (e.g. changing acceleration conditions)

. Archiving (e.g. beam parameters, profiles, settings, operation logs,...) processing and presenting.

Workstations assum human operator interface :

- Xterminal management for operator choice (task and hook names).
- Process server to run processes selected by the Xterminal.
- Translation algorithm to change hook names into hardware addresses.
- Local presentation of alarms.
- Beam control processes with MOTIF widgets as operator interface.

At equipment level

Controls are handed over to CAMAC/VAXELN and VME/VAXELN front end controllers. GANICIEL functions provided by these FEC are displayed on Fig 3.

Real time processes achieved at this level are :

- Communication servers to receive all the requests from workstations or other crate controllers.
- Hook to reserve and refresh the values of hooked pieces of equipment.
- Surveillance to check whether a piece of equipment is off specified limits.
- Alarms to send message to the main alarm server running on µVAX3800.
- Handlers to control piece of equipment with drivers that only control the CAMAC.

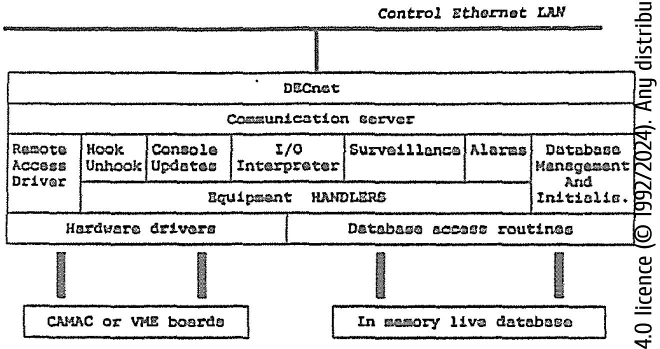


Figure 3. GANICIEL Functions on FEC

II. 3 Information management

Information management is of basic importance in a control system. Due to :

1. the various natures of information to manage :

- acceleration conditions, beam parameters (ion species, energies,...)
- realtime controls (node addresses, equipment identifications and characteristics, alarm messaging,...)
- operation logs
- reference characteristics of controlled parts (hardware installation and software)

2. the huge amount of controlled data (e.g. > 2500 pieces of equipment, beam parameters,..)

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S02SRU04

3. the distributed topology of our control system architecture (data flow vs time, up - and downloading, ...), information management has to be carefully analyzed and designed to meet performances (access, integrity, speed,...) and avoid conflicts which may be fatal.

Database management system

A careful investigation has led us to adopt a relational database management system (RDBMS) to fulfil our needs regarding NON-realtime purposes. We finally chose the commercial INGRES RDBMS as having specifications which fulfil our requirements and feature the open aspects we are looking for. Main reasons are listed as followed :

- Hardware independency, which allows straight forward integration in the VAX/VMS environment.
- Comply with SQL standard.
- Encompasss a 4th generation language in windowing environment (W4GL).
- High integrity, homogeneity of the products (QBF, ABF, ...) with good ergonomy consideration.
- ADA interfacing.
- and last but not least, a quality support.

Realtime database

Realtime database is hosted in the μVAX3800. It is fragmented into dedicated live databases reflecting the hardware configuration of the frontend CAMAC or VME/VAXELN local controllers. At these lower level locations, live databases are eventually reformated into appropriate structures for easy access by equipment handlers and for fast response, and reduced for memory saving. These live databases actually include static information and dynamic data about the operation of the controlled pieces of equipment or subsystems, under the management of a DBserver. Downline loading of live database may be a critical concern to be mastered.

II.4 Human interface

Human interface at GANIL integrates recent graphic enhanced processing units, namely VS3100-76 workstations and VT1300 X-terminals, to benefit from their color graphic high resolution capabilities and X-Window standard compliance. It is accomplished via operator consoles, supervision systems, and specific field graphic terminals.

Operator consoles

Operator consoles are installed in the main control room for centralized controls and along the accelerator for field controls and immediate interventions (e.g., the electronics backbone gallery and the experimental physics areas).

***Two control levels*:**

1. Elementary level for individual equipment controls, the computer control system acting as a sophisticated multiplexer. This level provides utterly standardized controls for all pieces of equipment, by turning reassignable knobs,

2. Higher level for complex and global controls involving many different types of equipment, calculations and displays. This level achieves fully customized interactive controls, by running tasks.

***Implementation*:**

Designation devices are currently trackball and mouse, to track and catch the graphic objects (task or knob image) on the screen of the X-Terminal.

Input device is keyboard to enter alphanum data, such as the name of a piece of equipment or an expected value.

Output device for control tasks is commonly the screen of the workstation for color graphics, associated with color print out devices.

Control devices are the popular reassignable knobs (shaft encoded potentiometers) which are used on the "one knob - one piece of equipment at a time control" basis. These shafts are grouped into four-unit module to achieve ergonomy and performances.

Pertinent alarm messages will be displayed on the console screens following a uniform presentation strategy with color coding to speed up interpretation. Operators can control specific actions from these consoles, such as activating the display of detailed DBMS messaging for inspection and diagnostics or acknowledging a specific or a whole class of alarms to clear the screens. Fig 4 shows an operator console unit.

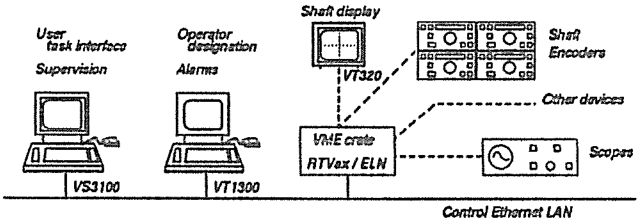


Figure 4. Operator Console Unit

Supervision

It uses worstations VS3100 to run the commercial software package IMAGIN supplied by SFERCA. This supervisory system allows control of any processor which is within the reach of the workstations, management and display of the collected data on animated synoptics over the background view. Supervised processors are currently programmable logic controllers (PLC) and the present control system. Imagin is composed of several software modules : a graphic editor, a configurer to create dynamic objects and an animator to perfom real-time display. Mailboxes provide communication threads between the animator module and the application process, as well as access synchronization to the shared data memory pool. PLC data acquisition is realized by the data server based on the SFERCA subsets PROLINK+ and its industrial database BDI (Fig 5).

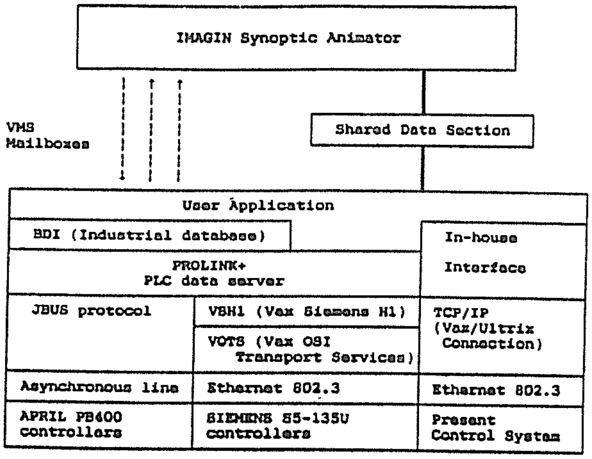


Figure 5. Supervision Structuration

Supervisory development environment is organized around wokstations linked to the VAX 6000-410 host processor, while supervision is performed on workstations related to the µVAX3800 cluster. ADA and FORTRAN application processes are interfaced to the various SFERCA modules through specific interfaces. Recent implementations encompass DECwindows environment. Examples of supervision will be presented in the following section.

Field graphic terminals

Programmable logic controllers are addressable by dedicated field terminals with attached keyboards to perform specific actions. These devices are for use by specialists for local inspection and expertise.

III. FIRST IMPLEMENTATIONS

III.1 Controls of the new ECR Ion source

A new 14.5 GHz ECR ion source, named ECR4, was installed on a 100 kV platforme to increase the beam intensity for metallic and heaviest ion species. This ECR source is controlled by a pair of SIEMENS S5-135U PLC. One PLC is on the high voltage platform and is linked by an optic fiber connection to its grounded companion. The grounded PLC is coupled to the Ethernet LAN to communicate with the supervision system running on a VS3100 workstation. An ADA application interfaces the IMAGIN software to animate synoptics. The ECR4 synoptics consists of several views to handle the RF power transmitter, to control the vacuum system and to drive the main parameters of the ECR source like the gas pressure of the UHF power, as shown on Fig 6.

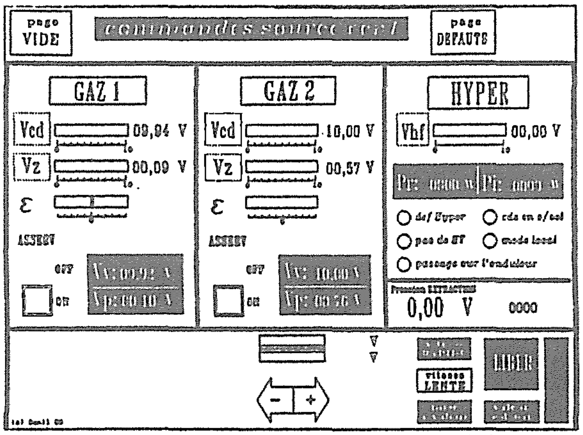


Figure 6. New ECR Ion Source Synoptic

III.2 Supervision of the internal temperature of the RF cavities

The temperature of water cooling the seven RF cavities of the accelerator are measured by the means of 34 PT100 probes connected to a SIEMENS S5-135U PLC and are supervised such as for the ECR source. In addition data related to beam characteristics, RF voltage and vacuum pressure are read from the present control system. Fig 7 shows the temperature measurements inside the northbound RF cavity of the first separated sector cyclotron (SSC1).

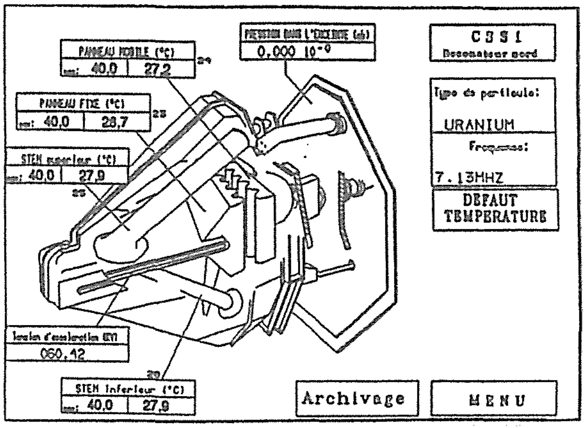


Figure 7. Temperature of a RF Cavity Synoptic

This application is the first experience of introducing relational database concepts into our controls environment. The INGRES RDBMS is used to archive measurements at operator request or in automatic mode, every 15 minutes. An off-line application using the Windows 4th Generation Language (W4GL) from INGRES displays graphs and presents result, depending on the stored data, as shown in Fig 8. It demonstrated the benefit that can be taken when developing with this language. It also led us to face the sensitive implementation of ADA/SQL interface and use of the ADA multitasking features in this case. The whole application is now about to be operational.

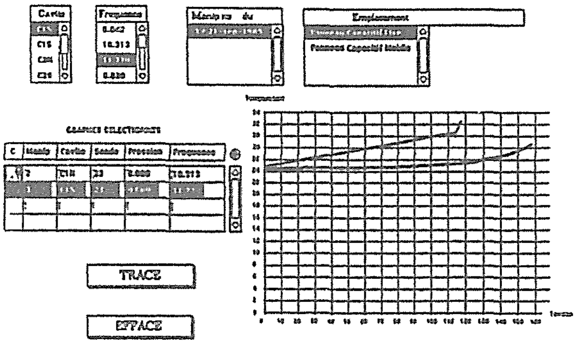


Figure 8. Off Line Display of a RF Cavity Temperature

III.3 Operation viewing and statistics

A W4GL application was written to store in an INGRES database the chronology of the GANIL operation. According to the screen displayed on Fig 8, the operator using a mouse has to point to or choose various graphic objects to specify what occurs while controlling the accelerator and the experimental areas. Every quarter of an hour, he has to indicate time and location of occuring failure, as well as beam tuning conditions and target experimental rooms. Later on, some of these manual operations will be filled up automatically.

The other part of this application to be developed will run on a Macintosh station. This application will read the INGRES database through the graphic query language (GQL) product to feed an EXCEL application for statics and report purpose. This application is planned to run by the beginning of the next year.

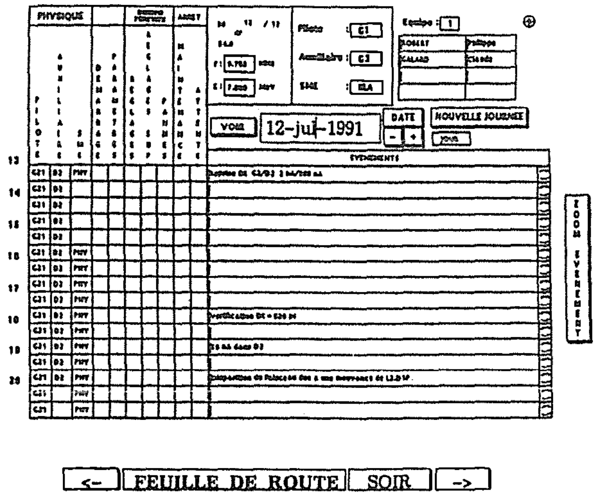


Figure 9. Operation Statistics Display

III.4 Beam profiler

It was important for us to validate by an example several real time control system basis components :

- Client server communication with workstations as clients and VME crates as servers.
- VME crates for the control of local pieces of equipment with data links to the control room workstation for display.
- The VME300 under VAXELN as crate controller.
- The DEC graphic editor VUIT to build MOTIF menus.

That was done with the beam profilers system in which the VME crate contains the 20 acquisition slots for 160 profilers.

On the VME crate, a communication server receives the requests from the workstations, according to the different allowed functions (gravity center, broken wires management, full width at half maximum). There is one ADA task for each function, so that simultaneous different requests can be satisfied. These functions find their data in an ADA task which is started by an interrupt at every end of the acquisition cycle.

After that, the computed data are sent by the LAN to the requesting workstation. An ADA program on this VMS workstation manages all the MOTIF widgets for the pop-up menus dedicated to the operator's choice, then displays and refreshes the graphic representation of eight beam profilers.

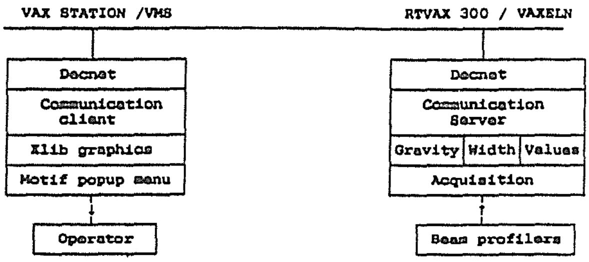


Figure 10. Beam Profiler Software Structuration

This beam profiler display application showed us that this kind of communication is fully satisfactory and can be extended to the fifteen other local control processes such as high frequency or beam phase management.

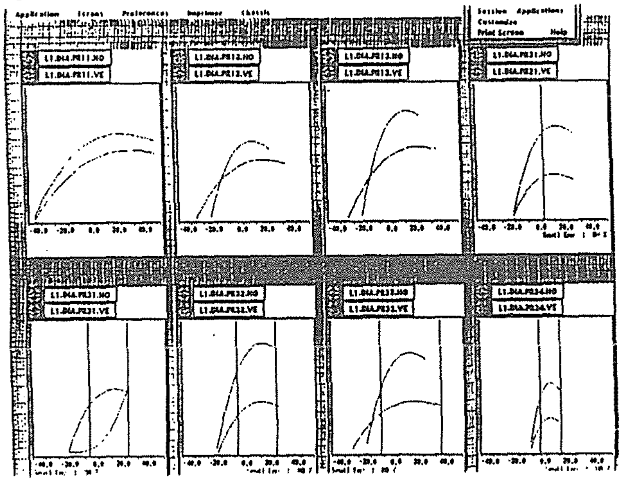


Figure 11. Beam Profiles

CONCLUSIONS

The host level is fully equipped and operating as well as the processors of the real-time control level : μVAX3800 cluster and workstations. The Ethernet LANs, to federate the three control levels are installed and provide satisfactory services.

Significant effort was deployed to analyze and build up the software architecture.

The next step will be mainly devoted to completing the GANICIEL specifications and to coding the system and user software.

The future control system is planned to run by spring 1993.

ACKNOWLEDGEMENTS

The control system described here is the result of continuous and combined effort involving others members of the Controls Group : P.de Saint Jores, E.Lemaitre, P. Lermine, C.Maugeais, F.Regnault, J.F.Roze, and our visiting guests : J. Galvez and G. Vega from the University of Bogota.

We benefit from discussions we continue to have with the people from the Electronics Group and the Operation Group of the Accelerator Operation Sector.

FURTHER READINGS

- E. Lécorché and the GANIL Operation group "A New Open GKS based Supervision System in use at GANIL" Proc. Int. Conf. on Accelerator and Large Experimental Physics Control System, Vancouver, Canada, 1989
- L. David, E. Lécorché, T.T. Luong, M. Ulrich . "The GANIL Computer Control System Renewal" Proc. Int. Conf. EPAC'90, Nice, France, 1990
- T.T. Luong "Real Time System for Local Intelligence" Europhysics News 22, 1991

REPLACEMENT OF THE ISIS CONTROL SYSTEM

R.P. Mannix, C.J. Barton, D.M. Brownless, J.C. Kerr

Rutherford Appleton Laboratory, DIDCOT, Oxon., OX11 0QX, U.K

Abstract

In operation since 1985, ISIS is the world's most powerful pulsed spallation neutron source. The decision has been taken to replace the existing ISIS control system, which has been in use for over ten years. The problems of such a project, given the legacy of processor specific hardware and software are discussed, along with the problems associated with incorporating existing interface hardware into any new system. Present progress using commercial workstation based control software is presented with, an assessment of the benefits and pitfalls of such an approach.

I. INTRODUCTION

ISIS is based on an 800MeV proton synchrotron, running at 50Hz, providing an average beam current of 120µA onto a Uranium target. The injector, synchrotron, extracted proton beam line and target station have been under control of the present system both during commissioning (1980-85) and operation. Work is in progress to attain the design current of 200µA within two years.

The control system (old CERN-SPS pattern), is based on 5 GEC computers, assembly language Data Modules, a general purpose multiplex system for equipment interfacing (MPX) , CAMAC based operator interfaces (see Fig. 1) and an interpretive control language. There are approximately 15000 lines of data module source code. The equipment interface consists of 700 modules in 70 crates. There are several hundred control programs in use.

The present control computers are very modest in power and are very limited in storage capacity. The operating system allows us to control hardware from a number of concurrent interpreter processes on each processor. High priority processes communicating with hardware completely lock out all others. Communication with other systems is non-existent, peripherals such as floppy disks are obsolete and unsupported, backups require shutting down the control system and maintenance is expensive (24hr cover is essential). The various branches of the interface hardware are tied explicitly to particular processors– reconfiguring after a processor failure is impossible– a serious disk drive fault can, and has, shut down the accelerator for a few hours. An average ISIS experiment lasts two days, within which several data taking runs, all of which are essential, must be performed. These sorts of breakdowns, although infrequent, are highly undesirable.

The new system is required to take over all the current functions of the existing control (at least as well) and have the capability for much increased data storage, greater reliability, easy reconfiguration and extension and almost transparent communications with other computer systems.

The available effort is about 25% of that when the original system was created, when both decreases in staff and the load of supporting the existing system are taken into account.

II. PROJECT PLAN

The system currently being developed is based on the Vista Controls[1] software suite, and the Hytec Electronics[2] Ethernet CAMAC Crate Controller (ECC). The new software provides a fully distributed database driven control system with a graphical interface over a number of DECnet nodes (currently VMS only although work is in hand on a POSIX-compliant version). Each control or monitoring object is referred to as a *channel* and, by using the *channel name*, control screens can be generated with an interactive draw package. Our current development system is based on two DEC VAXstation 3100 colour workstations and two DEC VT1300 X-terminals, although it is not clear that the choice of processors is optimal. Figure 2 shows a general arrangement of the proposed new system, based on Ethernet (an alternative transmission medium would be FDDI).

The use of channel names, database and handlers (equipment routines) maps very well on to our existing system based on Data Modules. The graphical interface provided with the new software should enable the functions of 75% of the control software to be replaced without recourse to writing code.

The system-wide nature of the databases and the networked nature of the CAMAC driver crates makes access to all equipment possible from any processor– something which was not previously possible.

Terminals with any level of access to the control system can easily be added anywhere on site (if desirable!). A control system for a new beam line can be provided by buying a workstation, another CAMAC controller and local interfacing hardware. The incorporation of this into the existing system is automatic (subject to the licensing agreement and an upper limit on the number of databases).

The current operator interfaces (touch sensitive screens , tracker balls, colour displays, knobs etc.) which are all obsolete or nearly so, will be replaced by high quality mouse/tracker ball and keyboard driven workstation type displays.

No new system will look or behave as the old one did. Users are familiar with the old system and will be reluctant to change. Any shortcomings in the new system will be picked on and amplified, shortcomings in the old system having been assimilated years ago. Use of the new system from a user-written program is more complex and less flexible– we are currently developing a simple interface to improve this..

There are three major parts to the management of change:

1. Transport of existing software.

2. Connecting to the new control system.

3. Training users

Most of #1 can be done off-line, that is to say without interfering with the operation of the accelerator, and this is currently in progress.

#2 is straightforward because all equipment below the dotted line on figure 2 is to remain the same. The CAMAC crates which drive our MPX branches merely have to have their crate controllers removed and replaced with the ECC controller modules. Changing the existing main control desk is highly problematic. Workstations will have to be installed side by side with the old control system on the first live runs, so that the old system can be reverted to if necessary. It must be stressed that ISIS runs as if it were a commercial enterprise, and our "customers" will not allow scheduled run time to be removed or interfered with for development purposes.

Training of staff, given our limited resources, is difficult. The operation of ISIS, although still to be further developed, could be said to be stable, so a straightforward functional replacement would be an acceptable first stage.

There is a long shutdown period of 3 months every year when it is expected that changes will take place.

The original time scale was for a two year project, culminating in a long shutdown. This is not feasible given the current manpower levels, so April 1993 is now the target date for a live run of the new system..

III. PROGRESS

Three databases have been written, those for the injector magnets, the injector timing system and the injector general purpose status modules– a total of 1200 channels so far. Handlers for the programmable timing modules, the most complicated of our magnet power supplies, and the status reading hardware have been written. We are preparing a specification for extra functions to be added to the Ethernet CAMAC crate controller, to reduce the overheads on equipment access. Control screens for the timer modules and magnet power supplies have been prepared and a suite of hardware test programs written for test access of interface modules via the ECC controller (external to the database).

Fig. 3 shows a workstation in use in the new control system. The menu windows running around the bottom right hand corner are replacements for our existing touch screens, where further menus and/or control screens are called up. The main part of the display is occupied by a control screen for the quadrupoles in the Low Energy Drift Space of the ISIS LINAC, showing control sliders and a monitoring strip chart. Fig 4 shows the operation of the control screen for the ISIS LINAC programmable timing system. No high level programming was required to generate these windows.

A significant amount of time has been spent deciding how to modify the standard usage of the Vista software to suit our needs and in moving to new versions. Now this has been done, production of software should speed up.

The choice of processor platforms on which to mount the system is not obvious. It is driven by the need to maximise perceived response time. We are not sure at this stage whether workstations are more or less appropriate than a powerful multi-user VAX running several X-terminals. Increases in workstation power may overtake this problem.

There is a lack of flexibility in hardware calls routed through the database, stemming from the inability to parameterize calls. One might wish to be able to retrieve the value of a single status bit from a 16-bit status port. In the present system, this is done by specifying the module, port and bit number– a 0 or 1 is returned. In the new system, to be able to randomly access any single bit in this manner would require database channels for every single bit, which would be very wasteful. We have opted to assign an integer channel to each 16-bit port, individual bit channels being set up on demand. On the other hand, this rigidness leads to a strong "typing" of database channels, minimising errors.

Complex control screens may be devised without any programming effort, however the hierarchical control over which screens may be displayed at any one time seems lack-ing to us and we are devising our own software to handle this.

Quick turn round support is available from the vendors by e-mail, phone or Fax.

On any physical Ethernet segment up to 256 CAMAC crates (each driving up to three MPX branches with up to 16 MPX crates per branch) can be simultaneously available from any processor with the ability to restrict access to individual modules to particular processors if required. The ECC modules have been reliable in operation with excellent support from the manufacturers.

The major difficulty with the ECC controller is an overhead (on single operations) of 10mS per transfer. The 10mS is a fixed feature of the VAX-Ethernet configuration and is not affected by transfer size. In a data acquisition environment the problem starts to disappear as the blocks of data transferred get larger. For a control system such as ours, where accesses to the hardware are much more random and multi-sourced, this is a large problem for which there is no clear solution as yet. The advantages of the ECC solution are the total flexibility and the overcoming of many of the limitations of CAMAC (to the extent of giving it a new lease of life).

IV. CONCLUSIONS

The choice of commercially available software must be correct for those establishments where lack of effort and staff turnover are problems. Good local support and constant updating of the product are also essential, as is the ability to feed requirements into the supplier's development plan. Even so the effort involved in changing to a new system is always under-estimated, both by the customer and by the supplier.

No commercial product will allow the easy assimilation of an existing control system. Whatever practises and methods prevail in an operating machine are the "right" ones by virtue of the fact that they are in use and familiar to those who use them. Any new system must be modified to fit what exists– not the best way to proceed. It is also clear that the only timing information of any interest to the user are the times taken to (1) present the control window required on the screen and (2) to operate a piece of hardware and see the effect on the screen, what happens underneath being irrelevant.

We are happy that the new control system will meet all our expectations with regard to extensibility, reconfiguration and communication and will perform as well as or better than the existing system. It seems to be an unwritten law of control systems that, as the power of the processors increases, the complexity of the software rises until the perceived response time drops to the minimum acceptable. We would hope that the extra complexity in the new system will be achieved without a drop in perceived response time.

The suitability of Ethernet (or any general purpose network system) as the main i/o channel is unclear. On the one hand emerging computer systems are frequently only supplied with an Ethernet port and moving away from this becomes expensive. It is truly distributed and configuring the system and expanding it become trivial. On the other hand, the generality of the system means that it is slow. It is clearly unsuitable for a fast data acquisition system but may be well suited to a supervisory control system such as ours. It allows the integration of PC's, CAMAC, VME, STEbus, and other systems in a controlled manner. Together with distributed software such as that described, it provides a system which is easily reconfigurable should a processor fail. In the ECC CAMAC controller itself there is also a large amount of untapped power although an easier way of accessing this would be an advantage.

REFERENCES AND ACKNOWLEDGEMENTS

1. Vista Controls Inc.
127 Eastgate Drive
#30800
Los Alamos, NM 87544
USA
Phone: (+1) 505 662 2484
Fax: (+1) 505 662 3956

2. Hytec Electronics Ltd.
Ladbroke Close
Woodley
Berks RG21 3YT
U.K.
Phone: (+44) 734 697973
Fax: (+44) 734 441068

(These companies are reciprocal agents)

GEC Computers are now GPT Datasystems Ltd.

DEC, DECnet, VAXstation 3100, VT1300, VMS are registered trademarks of the Digital Equipment Corporation.

Ethernet is a registered trade mark of the Xerox Corporation.

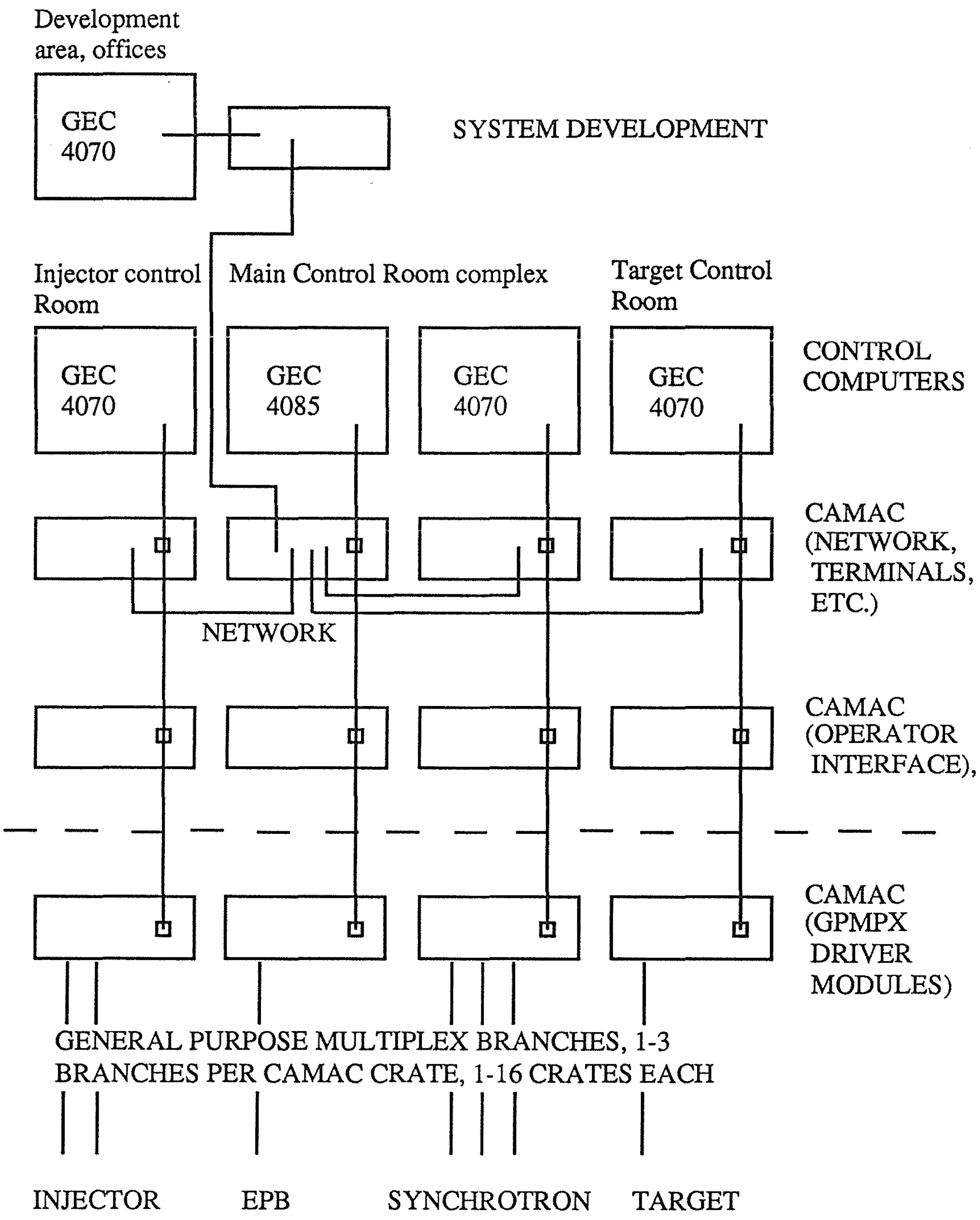


FIGURE 1. General arrangement of present system

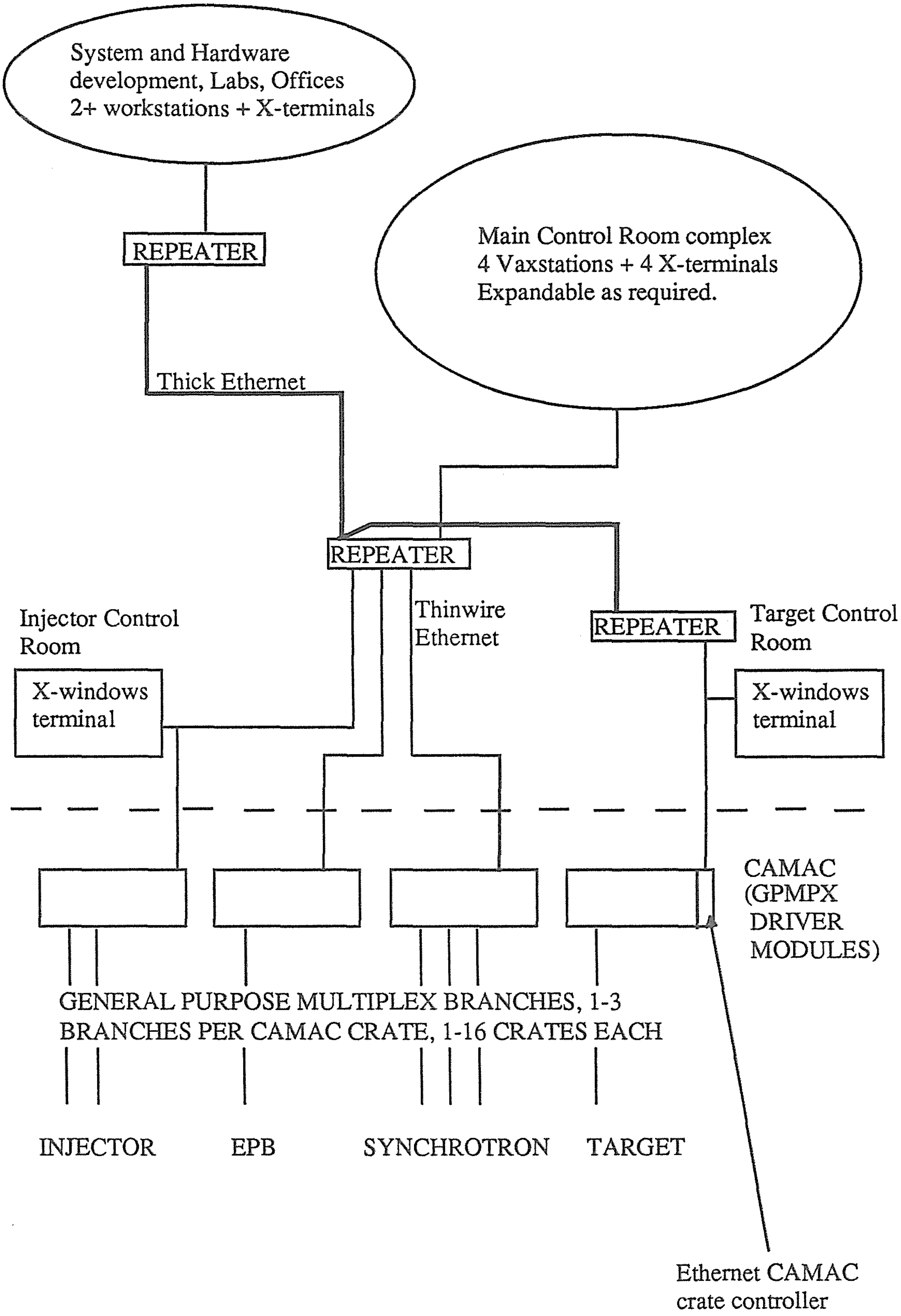
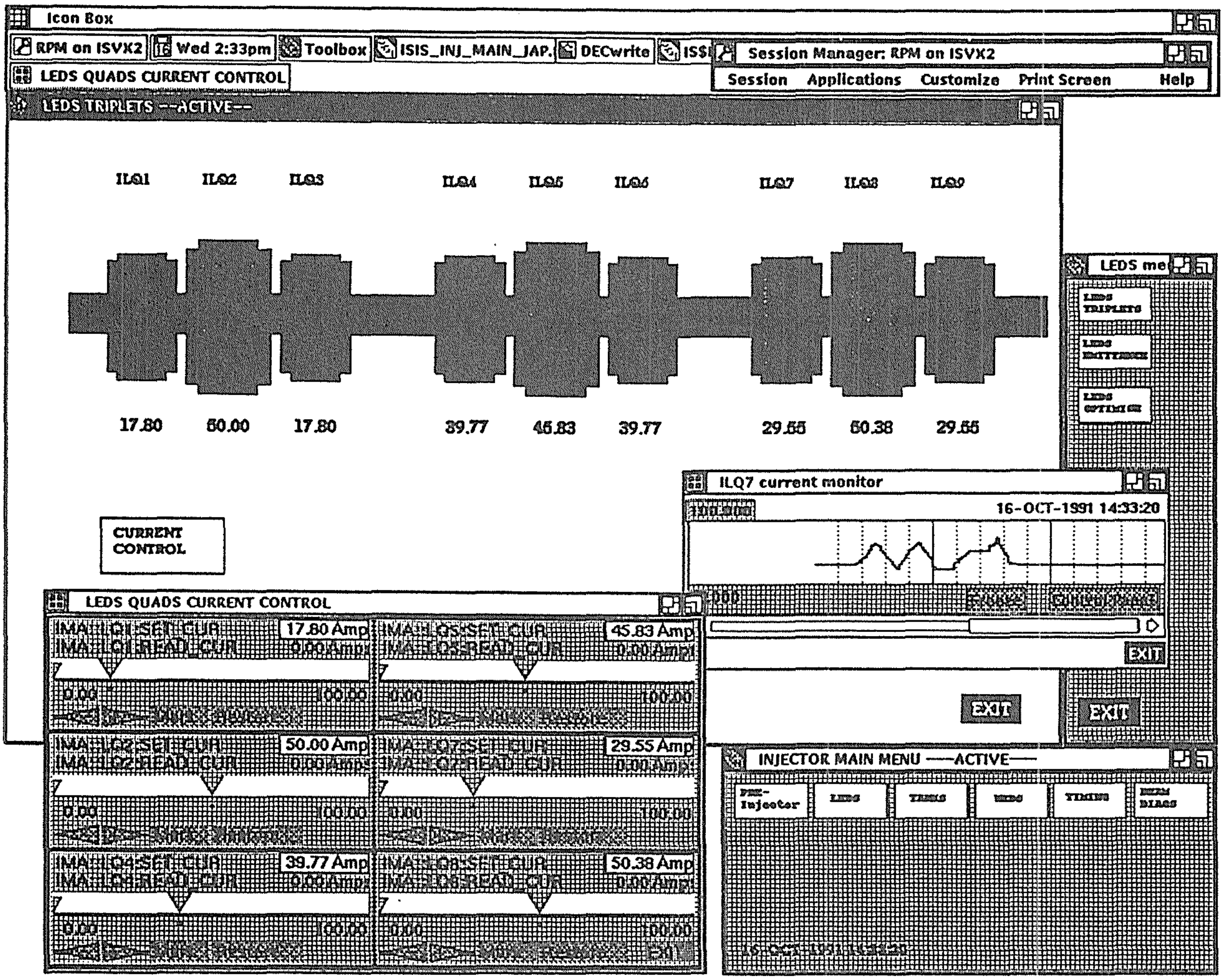


FIGURE 2. General arrangement of proposed system



17.80 50.00 17.80 39.77 45.83 39.77 29.55 50.38 29.55

16-OCT-1991 14:33:20

LEDS QUADS CURRENT CONTROL

IMA:ILQ1:SET_CUR 17.80 Amp
IMA:ILQ1:READ_CUR 0.00 Amp
0.00 100.00

IMA:ILQ5:SET_CUR 45.83 Amp
IMA:ILQ5:READ_CUR 0.00 Amp
0.00 100.00

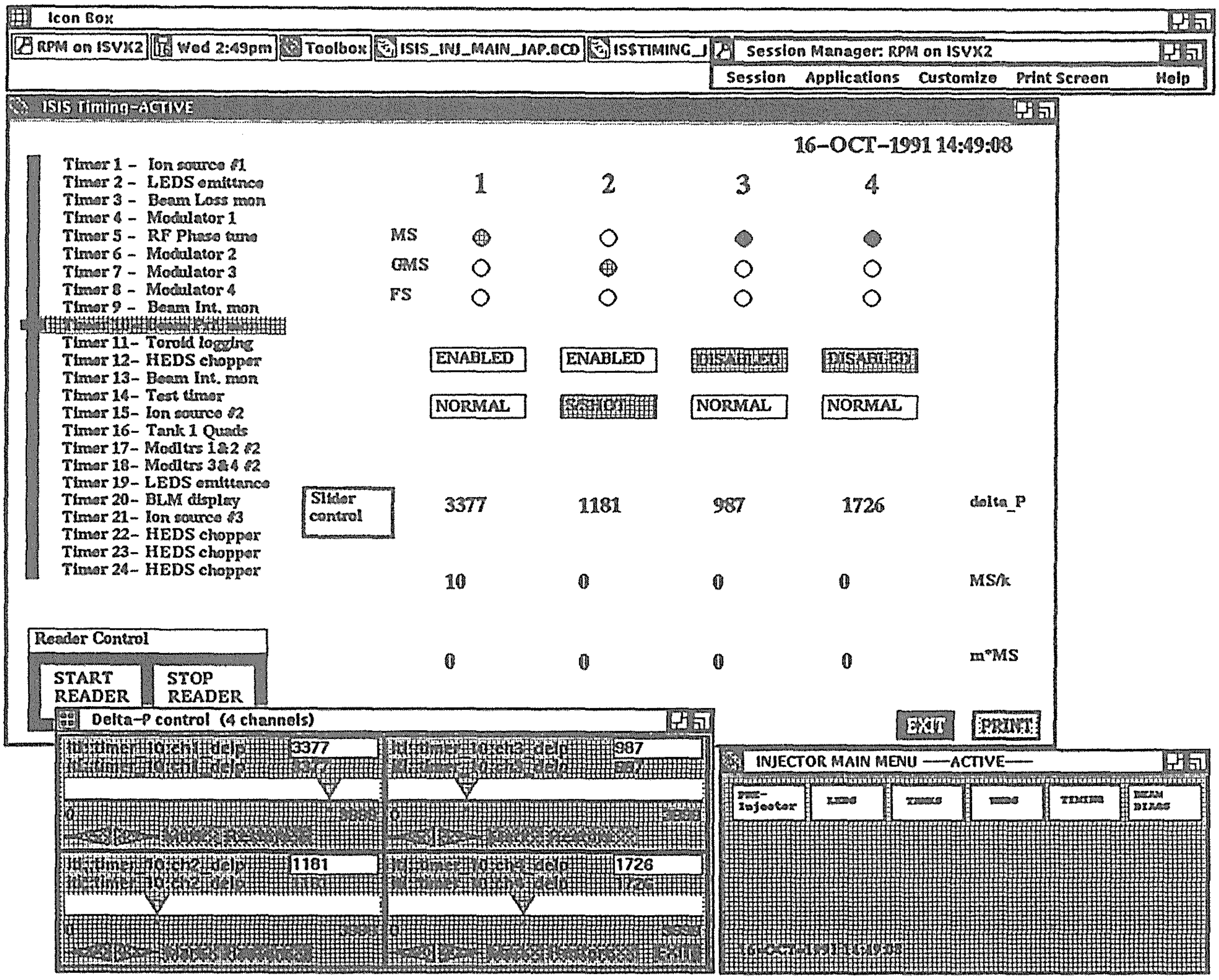
IMA:ILQ2:SET_CUR 50.00 Amp
IMA:ILQ2:READ_CUR 0.00 Amp
0.00 100.00

IMA:ILQ7:SET_CUR 29.55 Amp
IMA:ILQ7:READ_CUR 0.00 Amp
0.00 100.00

IMA:ILQ4:SET_CUR 39.77 Amp
IMA:ILQ4:READ_CUR 0.00 Amp
0.00 100.00

IMA:ILQ8:SET_CUR 50.38 Amp
IMA:ILQ8:READ_CUR 0.00 Amp
0.00 100.00

Figure 3: Example control screen– ISIS LINAC Low Energy Drift Space triplets.



16–OCT–1991 14:49:08

Timer 1 – Ion source #1
Timer 2 – LEDS emittnce
Timer 3 – Beam Loss mon
Timer 4 – Modulator 1
Timer 5 – RF Phase tune
Timer 6 – Modulator 2
Timer 7 – Modulator 3
Timer 8 – Modulator 4
Timer 9 – Beam Int. mon
Timer 10 – [illegible]
Timer 11– Toroid logging
Timer 12– HEDS chopper
Timer 13– Beam Int. mon
Timer 14– Test timer
Timer 15– Ion source #2
Timer 16– Tank 1 Quads
Timer 17– Modltrs 1&2 #2
Timer 18– Modltrs 3&4 #2
Timer 19– LEDS emittance
Timer 20– BLM display
Timer 21– Ion source #3
Timer 22– HEDS chopper
Timer 23– HEDS chopper
Timer 24– HEDS chopper

	1	2	3	4	
MS	⊕	○	●	●	
GMS	○	⊕	○	○	
FS	○	○	○	○	
	ENABLED	ENABLED	DISABLED	DISABLED	
	NORMAL	S/SHOT	NORMAL	NORMAL	
	3377	1181	987	1726	delta_P
	10	0	0	0	MS/k
	0	0	0	0	m*MS

id:timer_10:ch1_delp 3377
id:timer_10:ch3_delp 987
id:timer_10:ch2_delp 1181
id:timer_10:ch4_delp 1726

Figure 4: Example control screen– ISIS LINAC timers.

Upgrading the Control System for the Accelerators at The Svedberg Laboratory

K.J. Gajewski, L. Thuresson, O. Johansson
The Svedberg Laboratory
Uppsala University
P.O. Box 533, 751 21 Uppsala, Sweden

Abstract

Two accelerators at The Svedberg Laboratory in Uppsala, the Gustaf Werner cyclotron and the CELSIUS ring, will get a new control system. At present both the cyclotron and the ring have their own control systems based on S99 and PDP-11 minicomputers respectively. There are also a number of subsystems which are controlled separately from the stand-alone PC based consoles (ECR ion source, electron cooler, vacuum system). The goal of the rejuvenation is to integrate all existing control systems and provide the new system with an uniform operators interface based on workstations. The obsolete S99 microcomputers will be substituted with a VME system and all subsystems will be connected to the Ethernet. The upgrade strategy enabling the transformation of the system without any long shut-down period is discussed. Hardware and software planned for the upgrade is presented together with a discussion of expected problems.

1. INTRODUCTION

The control systems for Gustav Werner cyclotron and the CELSIUS ring have been designed at different times. The cyclotron [1] after a major rebuilding started its operation in 1986 The main cyclotron control system, based on S99 microcomputers, covers the cyclotron and all beam lines. The radiation protection system for the whole laboratory area is also based on S99 microcomputers with an 286/PC as a console. Another 386/PC connected to a Programmable Logic Controller (PLC) controls the external ECR ion source. The CELSIUS control systems is based on the LEAR control system from CERN slightly adapted to our needs. The system is based on a PDP-11/73 minicomputer connected via CAMAC hardware to the accelerator equipment. This system enables to control all function generators and all static parameters but these for the electron cooler. The electron cooler is controlled independently from an IBM PC. Some other subsystems, both on the cyclotron and the storage ring side, are also controlled from the 286/386/PCs. From the very beginning the work on these control systems was completely independent - there were separate teams for each machine (the cyclotron and the ring) and there was no cooperation between them. It all resulted in two distinct control systems with different philosophy and incompatible hardware solutions. After achieving the production stage of operation by the cyclotron and approaching the same stage by the CELSIUS ring a need for a common control system became obvious. Some organizational changes (creating one control group for cyclotron and the ring) should help to achieve this goal. The cyclotron and the ring is each controlled from its own control room with very limited possibilities to control or even only access the information from the other machine. This is highly unsatisfactory and the new control system will enable to control both machines from either of the control rooms. Some other features like beam sharing will be added. In order to improve the situation we plan to connect all substems together via an Ethernet and add a new user interface in the form of UNIX workstations. Some of the old S99 microcomputers will be substituted by a VME system running VxWorks. Such approach will enable us to achieve our objectives with minimal changes in the front end level and without any long shut down periods of the accelerators.

2. EXISTING CONTROL SYSTEM ARCHITECTURE

A. Cyclotron

The cyclotron control system is based on Texas Instrument (TI) microcomputers. At the top level there are three TI S99 microcomputers. The control system data base resides in one of the S99 systems, while the other two are mainly used as operator interface to the system. At the hardware level most of the equipment is connected to the control system by an in-house-built microcontroller called General Interface (GI). A single euro-board includes a TMS9995 microprocessor, analog and digital I/O channels, a terminal port for local control and a custom-designed serial communication bus. Up to 31 devices can be connected to this bus. Special interface boards (also in-house-built) are used to connect the bus to the S99 systems. There are about 150 GI's in the system. They are mainly used in the beam transport- and RF-system. Apart from the GI-controllers two industrial PLC units (Modicon 484 and 684) are used in the vacuum- and RF-system, fig. 1.

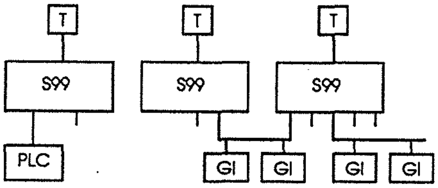


Figure 1. The main control system of the cyclotron.
T - terminal
PLC - Programmable Logic Controller
GI - General Interface (front-end controller)

The S99 systems runs TX990, a multitasking operating system from TI. Today both hardware and software support for S99 is very limited. Most of the control system software is written in assembly language. All this makes maintenance difficult and time consuming.

The radiation protection system also uses two S99 microcomputers. One S99 system handles the interlock and access control to the various experimental areas. The other monitors the radiation level by means of some 50 radiation detectors distributed in the laboratory. A 286/PC is used as operator interface. The radiation levels are continually logged to the PC disc. No distributed I/O (GI's) are used in this system, thus some 600 digital I/O signals and 50 counters are connected directly to the S99-systems, fig. 2.

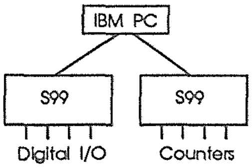


Figure 2. The radiation protection subsystem.

In 1989 an external ECR ion source [2] was added to the cyclotron. A Siemens S5 -115U [3] PLC is used as front-end. For operator communication a 386/PC is connected to the PLC system by means of a Siemens CP524 [3] communication processor, using an RS 232 channel to the PC. Most of the hardware is connected to the PLC but a few devices are connected directly to the PC using RS232, fig. 3.

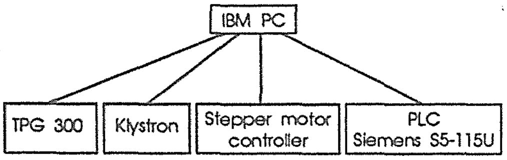


Figure 3. The ECR control subsystem.

MS-DOS was chosen as operating system for the PC, mainly because of the large amount of inexpensive software available for this environment and because the staff was already familiar with this operating system. Another alternative would have been to use a real-time multitasking operating system. This would probably have resulted in a more efficient system but at a higher cost. The present solution was considered adequate since the ion source control handles a relatively small number of parameters (<100).

The control program is written in Microsoft C. To simplify programming, a library of process control routines was written. This library allows creation of multiple threads of execution within a program. Threads can communicate through shared variables or message queues implemented in the library. The threads are scheduled using a nonpreemptive round robin algorithm. This eliminates the need for mutual exclusion mechanisms when accessing shared data. Scheduling only occurs when a thread makes a blocking call to the library (for example reading from an empty message queue). Driver routines for the RS 232 channels are also included in the library. This enables the driver to block a thread waiting for an I/O operation to complete.

B. Storage ring

The control system of the CELSIUS ring covers all function generators (25) and static power supplies (control and/or acquisition) in the ring (about 100), a vacuum system of the ring [4] (about 350 digital inputs/outputs plus 16 analog inputs) and the timing system. The whole system consists of three parts: main control system (fig. 4), electron cooler control (fig. 5) and vacuum control.

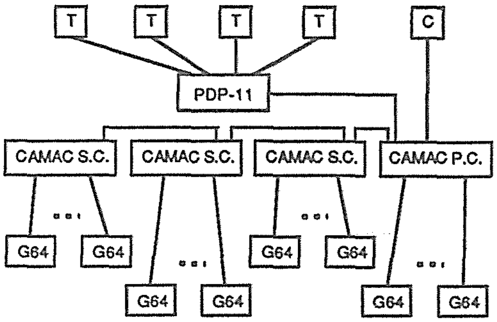


Figure 4. Main control system of the ring.
T - terminal, C - console,
CAMAC PC, CAMAC S.C - parallel and serial crate
G64 - front-end controllers in G64 standard

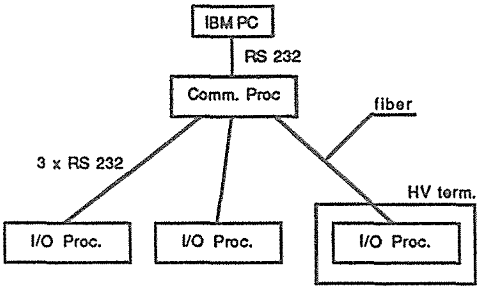


Figure 5. Electron cooler control.

The front-end level in both the main control system and the cooler control system is based on the G64 standard. The vacuum control is executed by a programmable logic controller (PLC). The PC in this case, connected to the PLC via RS 232 serial line is used only as an operator console. The PDP-11 is running under RSX11-M V4.2 operating system and DECnet. The kernel of the control system and many applications are written in MACRO-11 assembler. Other application programs are written in FORTRAN and Pascal. The PCs are running under MS DOS and the control programs are coded in Turbo Pascal. There are several data bases and data base management programs. In the main control system there is a database for static parameters, for dynamic parameters (vector tables) and for the timing system. The cooler control system has its own data base. All these data bases have different structure and contains various fields needed for the control tasks. The operator interface is also different for each system. The main control system have a console with a touch screen, numerical keypad, knobs and a semigraphical color display. The touch screen menus have a tree structure with 16 buttons on one page. The buttons can be used to pick up the parameter for control, to start an application program or to choose another page. A number of application programs have a text oriented user interface and are run directly from the terminal. The user interface of the electron cooler is based entirely on the PC working in the text mode. The operator can display up to 20 parameters on the screen. The absolute and incremental control is possible from the keyboard. Commands to the control programs are invoked by the function keys. The vacuum control system has a graphical user interface. The status of the vacuum system is presented on the screen with dynamic symbols (changing color or/and shape) on top of the static picture. The operator communicates with the system with use of dialogue boxes, buttons, text input fields etc.

3. UPGRADED CONTROL SYSTEM

A. Hardware

The hardware configuration of the upgraded control system is presented in figure 6. All control computers will be connected to a segment of the Ethernet which is isolated by a LAN bridge from the university network. A repeater must be used because of segment length limit. The front-end level of the control system will not be changed with an exception of a limited number of low level controllers which should enable switching between several control values synchronously with the ring acceleration cycle. This modification is going to support beam sharing between the storage ring and other cyclotron users as well as switching the settings of the electron cooler between injection energy and flat top energy. The switching will be triggered by external pulses from the timing system.

The changes in the hardware of the next level - local process computers, will be more significant. The S99 systems in the cyclotron control will be replaced by a VME system. A Force CPU-30 [5] will be used as local controller. Due to the large installation base of the GI-controllers they will be kept in the upgraded system. A new version of the GI-communication bus interface, adapted to the VME-bus, have to be manufactured in-house. RS 232 and Ethernet interface are included on the CPU-board. All PCs will be supported with the Ehternet controllers. An upgrade of some PCs based on the 80286 processor to 80386 with at least 2 MB RAM is necessary.

The radiation protection system will keep the S99 system as hardware interface since extensive rewiring will otherwise be required.

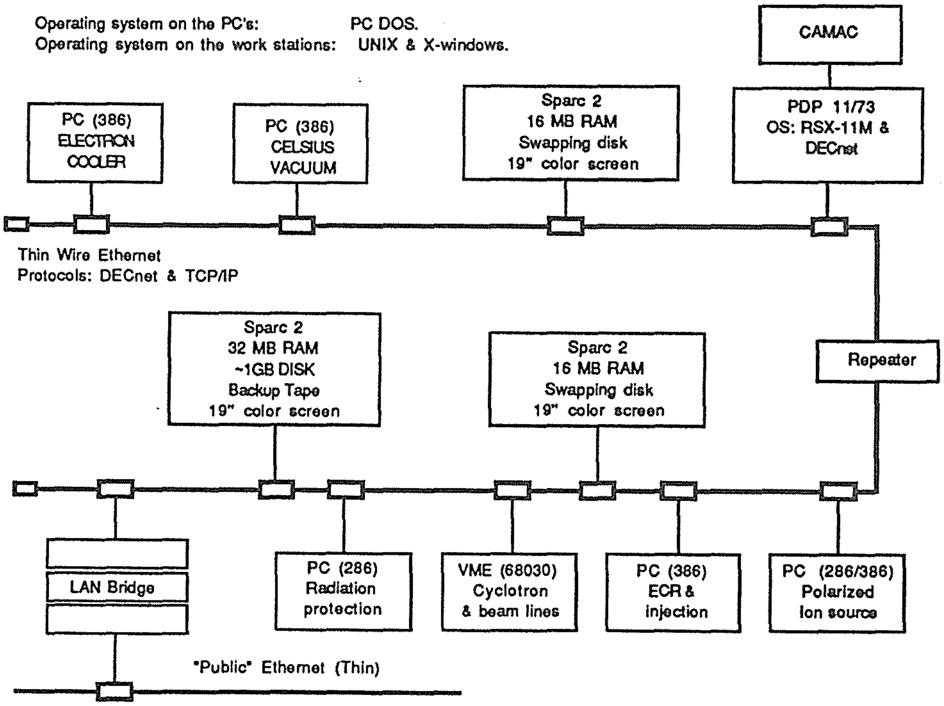


Figure 6. The layout of the upgraded control system.

An additional external polarized ion source [6] will be installed in the beginning of 1992. The ion source is bought as a "turn key" system including control system from Balzers/Pfeiffer. It uses a Siemens S5-135U [7] PLC as front-end and a 286/PC for operator communication.

The top level (user interface) will be based on Sun Sparc 2 workstations equipped with 19 inch color screens. We assume that the mouse and the keyboard will support satisfactory operator interface.

B. Software

The main effort in upgrading the control system lies in the software. Our strategy is to preserve as much of existing investment in the control software as possible. However, a considerable part of the programs will have to be rewritten or moved to another platform. Most of the programs on the workstation must be written from scratch. We have chosen TCP/IP as a communication protocol between the workstations, PCs and VME system. For communication with the PDP-11 we will use DECnet.

The VME system will run the VxWorks[1] real time operating system. The data base will be similar to the ECR ion source data base structure, with the addition of support for multiple control values to a parameter to enable "beam sharing". VxWorks is a "UNIX like" operating system. This means that program modules can be written and roughly debugged on the UNIX workstation, then recompiled and loaded in the target system.

To minimize the use of the poor software development environment in the S99 system, the radiation protection control program will be moved from S99 to PC level. The S99 will only be used for transfering the system I/O image between the PC and the actual hardware. The I/O image mapping server in the S99 will be unaware of the PC application program structure , thus enabling changes in the application without modifying the S99 software.

The programs running on the PC will be modified in such a way that they will be able to get requests from the network and send replies to the callers. The socket interface will be used. The network software we use (PC/TCP for DOS) can support up to about 20 open TCP connections. Most of the connections will be used by application programs and one is reserved for alarm and warning channel to the alarm server on the workstation. The existing programs should still perform all functions as without the network interface. This should allow to gradually move control from the PC to the workstations when the relevant programs on the workstation will be ready and thoroughly tested. We plan to use the same communication procedures and algorithms on all PCs. The programs written in Turbo Pascal will be rewritten to MS Pascal to enable easy linking to the PC/TCP network library.

The polarized ion source uses Wizcon [8], a commercial control system software package. This product have built in support for remote control nodes, using the Netbios communication protocol. This will present a problem since the control system network uses only TCP/IP and DECnet.

The data-base in the workstation will contain only information needed by the application program to access the parameter. All low level internal characteristics will be hidden to the application. There will be no global "life" data base with all parameters continually updated on the workstation. The application programs will get this information from the lower levels on request. The alarm server will listen to the information coming from the different nodes and inform the user of possible problems. This task should also check if all nodes are alive.

We will use SL-GMS [9] as a tool for creating the user interface. The GMS is a software tool kit for building dynamic graphical 2D man-machine interfaces. It was chosen mainly because its flexibility and good compatibility with the X environment.

4. SUMMARY

The work on upgrading the control system has begun about 6 month ago. During this time the requirements for the new hardware and software were specified. We stressed the necessity of using commercial, wide spread and well supported hardware and software. Most of the hardware is already in place or ordered. All computers have been connected to the Ethernet and some tests with the network communication programs have been performed. At the moment we work on the structure of the high level data base and communication protocols. The first program which we plan to write is a synoptic program which will show the status of any part of the accelerators and enable to control selected parameters. Next the application programs from the PDP-11 running on the console will be moved to the workstation. At the same time the work on moving the main cyclotron control to the VME standard will start. The programming and testing will be done on a development system not connected to the cyclotron. After positive tests the system will be installed on the target VME system. The alarm server, the archive program must also be written as soon as possible. As a last step we consider to move most of the application programs running on the PDP to the workstation. Most of them are terminal oriented, so in the transition period we can run them in the terminal emulation window.

The main problem we have to face in upgrading the control system is the incompatibility of existing data bases. This we plan to overcome by creating the new, high level data base for the programs running on the workstation, common for all parameters independently of their physical location. Another problem we meet is the limited pool size in the RSX-11 operating system. This limits the number of tasks which can run on the PDP and special attention must be paid to programming network servers, allowing only one instance of the task. We expect some problems with the RAM size limit imposed by the MS DOS, especially if we would like to keep all the functionality of the existing programs running on the PCs.

5. REFERENCES

[1] S. Holm, Proc. Int. Conf. on Cyclotrons and their Applications, West Berlin, May 8-12, 1989
[2] C. Ekström et al. "Status of the ECR ion source at the Uppsala cyclotron", 4th Int. Conf. on Ion Sources, Bensheim, Germany, 1991. (To be published in Rev. Sci. Instr.)
[3] S5-115U Programmable Controllers, Siemens Catalogue ST 52.3.
[4] K. Gajewski, "The Vacuum Interlock System for the CELSIUS Ring", *Nucl. Instr. and Meth. in Phys. Res.* A293, 1990, pp. 183-186
[5] CPU-30, Force computer hardware data book 1991.
[6] C. Ekström et al. "External Ion Sources at the Uppsala cyclotron", *Rev. Sci. Instr.* 61 (1990), pp.628-630
[7] S5-135U Programmable Controllers, Siemens Catalogue ST 54.1.
[8] Wizcon , PC Soft International, Wizcon User Manual 2.1
[9] D.G. Shelef, "Developing Graphics in Realtime", *SunTech Journal*, Vol. 3 No 3, pp. 64-74, July/August 1990

---

[1] VxWorks is a trademark of Wind River Systems, Inc.

Upgrading the BEPC Control System

Yang Li-Ping, Wang Li-Zheng, Liu Shi-Yao
Institute of High Energy Physics, Academia Sinica
P.O.Box 918, Beijing

Abstract

The BEPC control system has been put into operation and operated normally since the end of 1987. Three years's experience shows this system can satisfy basically the operation requirements, also exhibits some disadvantages araised from the original centralized system architecture based on the VAX-VCC-CAMAC, such as slow response, bottle neck of VCC, less CPU power for control etc.. This paper describes the method and procedure for upgrading the BEPC control system which will be based on DECnet and DEC-WS, and thus intend to upgrade the control system architecture from the centralized to the distributed and improve the integral system performance.

1. The system status and its milestone

The project of BEPC control system was determined to adopt basically from the SPEAR control system in january of 1985. The prototype control system was constructed during half year begun from september of 1985 at SLAC. In the end of 1987, the control system realized the on-line control and monitoring for the most equipments of BEPC Storage Ring (SR) and its Transport Line (TL), and was put into precommissioning at this time. The beam orbit correction has been brought into commissioning in june of 1989. The RF on-line control (including its ramp) was completed in march of 1990. Thus all the equipments of SR and TL were controlled by the computer, and the VAX-11/750 computer (the central control computer) becames a member of DECnet of IHEP at the same time.

From the operation experiences in the past several years, it seems that the philosophy and criterion for constructing the BEPC control system are available, which resulted that this system can be completed just on time schedule and have a high reliability of system operation.

2. Several problems in system

The BEPC control system architecture follows the old centralized control model of the SPEAR, we had done incessantly some improving work on system level in the past years, but still there are several problems which can't be solved. These problems are as follow:

1. The unique VAX-11/750 computer is used for the central control computer, its poor CPU power (only the 60 percent performance of VAX-11/780) limits the processing speed of whole control system; Many batch jobs (about 16 control processes) always reside in the memory, morever which heavily increases the load of VAX-11/750 computer system. These causes slow the response and processing speed of the entire system.

2. Due to we adopt the VAX-VCC-CAMAC (VCC, that is, VAX CAMAC Channel) hardware system architecture, one VAX QIO in CXCAMAC program takes 20 ms at least, so the VCC forms the bottle-neck of the control system communication. Especially it is obvious during the power supply ramping, the other jobs can not be serviced quickly.

3. The current human-machine interface is supported by two graphic colour monitors (resolution 512 × 512), two touch-screen with several computerized knobs. Now one monitor is occupied by one picture once a time, if we want to see several pictures of the different requirements of accelerator commissioning simultaneously, it needs to add more graphic display.

4. A fatal failure of the VAX-11/750 computer in the night (no on call service for computer) will break down the collider's operation. So a backup computer is needed to be considered.

3. System upgrade

After we analyzed the present conditions, we decided to reform the control system from the current centralized system to the distributed system and intend to solve those problems for improving the whole system performace.

a. The distributed control system architecture

In order to reduce the heavy load of VAX-11/750 and increase the speed of whole system responses, we divide the current control jobs on VAX-11/750 to several parts and load them into different computer nodes. the upgrade system will be based on the DECnet, its system architecture is shown in figure 1. The new added VAX-II computer is used as a intelligent node dedicated to the power supply control. The current VAX-11/750 also is a intelligent node which mainly is used for other subsystems, such as BPM, RF, VACuum. In the first phase, we will still use the current human-machine interface as the system console. In the second phase, we will add another DEC-WS as upgrade system console which can utilise fully the rich DECwindow's software functions and make the multi-display for improving the environment of human-machine interface. Some of the jobs involved a lot of calculation will be moved to the DEC-WS. These three nodes will be networked into a individual local network which is linked to current IHEP DECnet via a network bridge.

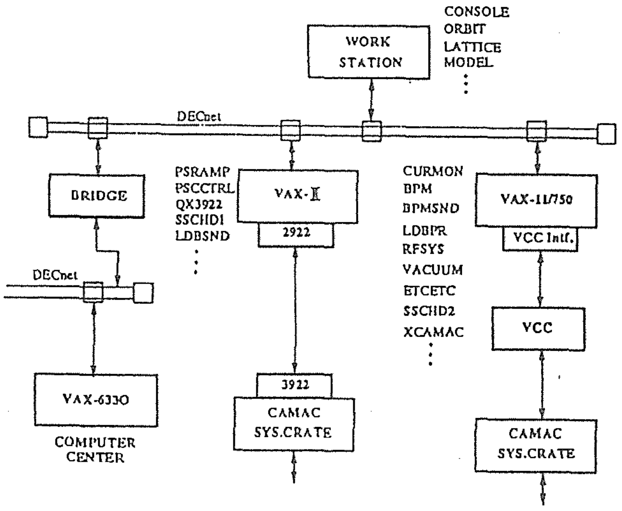


Figure 1. The upgrade of BEPC control system

b. Adopting Q-bus CAMAC adaptor

In the current system, all the datum are passing through the VCC, one QIO of VMS for VCC need about 20 ms at least and obviously reduce the processing speed of whole system. Beside the VCC adaptor is an old SLAC dedicated product, not a commercial module. Therefore, we adopt the Kinetic Systems Corporation's KSC 2922 and 3922 commercial modules as the VAX-II CAMAC adaptor. Under the support of KSC's, a VAX QIO operation only takes 3 ms in 1,000 words under the VMS, it would overcome the bottle-neck of the VCC and also increase the system reliability.

c. The compatiblity on new and old system

For keeping the compatiblity of CAMAC system, we are not going to do any modification on the hardware below the CAMAC system crates as far as possible. Thus after the upgrade is completed, the current control system on the VAX-11/750 can become the backup on Micro VAX-II. Conversely, if VAX-11/750 computer is out of work, the collider operation can keep continuously due to the power supply still be controlled by the VAX-II machine.

4. The upgrade of BEPC control software system

The upgrade work of the control system is mainly focused on the software side, the block diagram of system software upgrade is shown in figure 2, which can be divided into the following three parts:

1. Creating new control software system for power supply control on the VAX-II.
2. Remoulding the current control software on the VAX-11/750.
3. Coordinating the relations between VAX-II, VAX-11/750 and DEC-WS for integrating whole distributed control system.

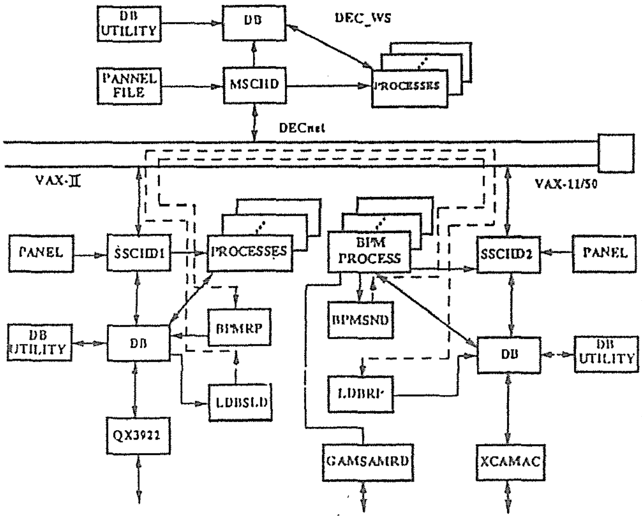


Figure 2. The upgrade of BEPC control software system

a. Upgrading procedures

First, we copy and modify a second set of BEPC control software system and its VMS operating environment on the VAX-II machine, and also modify and create the real time database (DB) on the VAX-II and DEC-WS as like as on VAX-11/750. Under such arrangement, we can reuse same panel files and all of the current application programs, reduce the software work to minimum and keep all the operator procedure as before.

We need to develope some communication programs for exchanging the necessary messages between three nodes on the DECnet. These programs are: a.) The schedule communicating program —— used for transferring the schedule control and processing message (as position coordinate of touch panel or cursor on DEC-WS later, knob datum etc.) of console node (either VAX-11/750 or the DEC-WS later) to other process nodes. b.) The power supply DB refreshing program —— the real time DB of power supply is resided on VAX-II, this program is used to send and receive all of the power supply DB parameters form VAX-II to the console computer for refreshing the DB on the console node continuously in 1 time/second rate. c.) Other communicating programs —— used for transferring the control and the relative information of BPM, RF and VAC subsystem between VAX-11/750 to VAX-II and DEC-WS.

We prepare to use the DEC GKS graphic packets or the UGS graphic system of SLAC to upgrade the current graphic programs. In order to remain and keep the current operating environment for operator and reuse all of the application programs, we want to save all the previous subroutine names and only change the contents from old graphic program to new upgrading graphic program.

According to our current upgrading police, we still want to keep and save all the current programs on each node. Therefore, we introduce a "Pseudo-Subroutine Method" (PSM) in the programs, i.e., if one process is needed to be activated, only the process resided on the active node is functioned, the same process on other nodes keep quiet without any activity. This PSM can reduce the upgrade work amount and short the upgrading time. Of course, it will introduce some disadvantages, such as, occupy some memory resource etc, which will be considered to improve it at later.

During the second step, when the DEC-WS will be used as the system console, we will transfer some application programs used for modelling-based control, such as Lattice, Modelling and Orbit correction processes to the DEC-WS later. Thus we can construct our minimum distributed system architecture under our tight budget condition and establish a basic distributed system environment for later development.

b. Some progresses at present

So far, we have got some progresses in the upgrade of software on the VAX-II during this summer shutdown of BEPC. In order to run the BEPC control software on the VAX-II, we have modified and changed some parameters and commands procedures concerned with BEPC application system resulting to optimize the environment of the upgrading control software; We have created the real time database and have debugged all of the DB utilities on VAX-II; Now, the process and subtask can be submitted and activated by the schedule (SSCHD1) process through calling the panel files which is same as on the VAX-11/750 while we touch the current touch-screen; When the BPM process on the VAX-11/750 is once activated, the BPM buttons datum can be acquired individully by CAMSAMRD program instead of the XCAMAC. The BPM process also calls BPMSND program to active the remote process BPMRP on VAX-II and send the latest BPM datum to it via the DECnet, thus the BPMRP refreshs the BPM area of DB; The status and parameters communicating process of power supply and the graphic program substitution which is based on the work station also got some progresses.

Acknowledgements

This upgrade project is based on the proposal which is presented by prof. Shi-Yao Liu in 1988, and supported by IHEP and I&C division. The collegues of the control group make their efforts for this upgrade now.

References

[1] Shi-Yao Liu and BEPC control group, BEPC computer Control System, presented at "Proceeding of the 14th International Conference High Energy Accelerators", August 22-26, 1989, Tsukuba, Japan. Page 899.

[2] Shi-Yao Liu, Accelerator Control for Beijing Electron Ring and Its Experience, presented at "The 14th HiSOR Meeting", Hiroshima University, Japan, March 5-6, 1990

[3] S.Howry etc. Portable database driven control system for SPEAR, Report on 1985 particle accelerator conference, Vancouver, B.C., Canada

[4] R. Appalaraju and Robert T. Cleary, VAX-Based Realtime data acquisition for Scientific Measurements application, Report, NSS, Orlando, Florida, Nov 9-11, 1988

The rejuvenation of TRISTAN control system

November 1, 1991

T. Mimashi, J. Urakawa, S. Kurokawa, .T. Kawamoto, S. Takeda,
A. Akiyama,. K. Kudoh,. K. Komada, T. Naitoh

National Laboratory for High Energy Physics (KEK),
Tsukuba, Ibaraki 305, Japan

abstract

The current TRISTAN accelerator control system uses CAMAC as a front end electronics, and they are controlled by twenty five Hitachi minicomputer HIDIC 80's which are linked with an N-to-N token ring network. After five years from now, these computers must be replaced. This is because of the life time of control system and we have to cope with the requirements imposed by our future project such as the KEK B-Factory and the main ring photon factory projects. The rejuvenation of this control has to be done under some constraints such as the lack of manpower, limited time and financing. First we review the problems of current control system, then the philosophy of the new generation control system is presented. Finally it is discussed how to move to the new generation control system from the current TRISTAN control system.

1.Introduction

Eight years have passed since the current control system started the operation of TRISTAN accelerator [1] [2]. We have a few weeks hardware maintenance for every six month to keep our system highly reliable. The Hitachi company takes care of these maintenance; they clean up each computer, check its operation and replace some parts such as fans, filters, etc. The disk of each computer is replaced every five years. Hitachi also supports 24 hour "on call shift" for any computers troubles. Supported by these maintenances, the current system has been working very stable for these eight years.

Recently some of the I/O devices become out of date, and it becomes difficult to repair these devices. In addition to the problem, the current system can not satisfy the increased requirement of accelerator control. The latter is mainly caused by the lack of CPU power and by the fact that the process computers are all 16-bit machine. And the network transfer capability is limited by the lack of CPU power. In order to solve these problems, it is about a time to start thinking about rejuvenation of TRISTAN control system.

2.The TRISTAN complex and its control system

The accelerator complex of TRISTAN consists of three major accelerators. A 400 m electron linac accelerates electrons and positrons up to 2.5 GeV and injects into the accumulation ring. The accumulation ring accelerates them up to 8 GeV and injects them to the main ring. The main ring is an electron-positron collider . Two electron bunches and two positron bunches are circulating in the opposite direction and collide with each other at the mid-points of four straight sections. The current experiment runs at the center of mass energy of 29 GeV [3].

There are two independent control systems for their operation, one of the system takes care of the operation of linear accelerator and the other takes care of that of the accumulation ring and the main ring. This paper discusses about the rejuvenation of the later system [2].

The TRISTAN accelerator control system uses twenty-five minicomputers and two large general-purpose computers (The main frame: Hitachi HITAC and Fujitsu FACOM) [4].

The minicomputers control hardware equipment and serve for man-machine communication. The general purpose computers are used for the calculation of closed orbit distortion correction which overload the minicomputers. The 25 distributed minicomputers, Hitachi HIDIC 80's, are connected by optical fiber cables to form a 10 Mbps N-to-N token-ring network. From each minicomputer, a CAMAC serial highway is extended to the hardware equipment.

The minicomputers are classified into two groups: the system computers and the device-control computers. The system computers control six operational consoles and two alarm-processing. The device-control computes are three minicomputers for RF cavities control, five for magnets control, two for vacuum controls, two for beam monitor, two for beam transportation, one for timing control, one for program development, one for a gate way between the control system and main frame and one for general purpose.

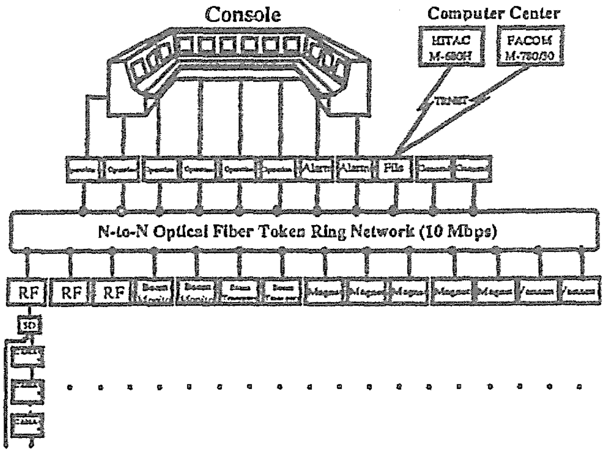


Fig 1: The current TRISTAN accelerator control system

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S02SRU08

Accelerators are operated through six identical operator's consoles which consists of two high-resolution color graphic CRT terminals and a pair of touch panels and a VT100 terminal used for a program development.

In the TRISTAN control system, the NODAL system was chosen as an application environment of the system. NODAL, a multi-computer language devised at CERN SPS, has been implemented on HIDIC 80's with enhancements such as a fast execution speed, a dynamic linkage scheme for external subroutines, etc. NODAL has multi-computer commands which enable us to transmit NODAL program from an operational computer to a device-control computer through the network and execute it on the device-control computer. All high level application programs are coded in NODAL and low level subroutines called as the data module are written by the PCL language that is the FORTRAN like language on HIDIC.

3.The plans of rejuvenation of the control system

The basic design of current control system was made by KEK and built by the company in the beginning of 1980's. Since there weren't any good standard products in that time, the proprietary system was built using their own network and process computers. And they developed software tools under their special environment. For example, NODAL is developed using PCL language. This is one of the typical case to build the control system in the industry.

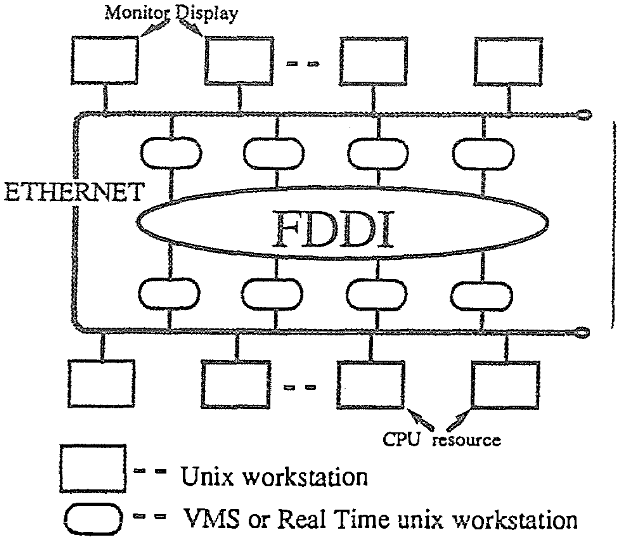


Fig 2 (A) : The possible new control system for TRISTAN using the real time process computers

Some of the industries order the other company to make full control system which includes their special application programs. Once their system was established, there is no change in the application program. People in the industry only takes care of ' Turn Key Operation '. The difference between our control system construction and that of industry is that people in the accelerator department wrote all application programs, and the number of programs has been increased.

Using this kind of relationship to the company, we could build the high quality control system in the short period with small number of staffs. The staff could concentrate on the application program development.

But it results in the single vendor system and the low degree of standardization, for example the system uses special network and special PCL language so that it causes a problem when we try to add the new technology to the system.

The new system should have high reliability, high flexibility, high maintainability, high performance, high extendibility, real time response and at the same time it is important to use multi vendor computers and the standard network protocol. In order to establish this system, the system should be simple and well modularized from the point of view of hardware and software.

The system architecture

The current system consists of two layers. One is the front end layer which is composed of about three thousand of CAMAC modules. And the other is composed of consoles and process computers.

The first step of the system rejuvenation does not touch the front end layer at all, so that only the computing environment will be replaced.

The distributed CPU's and the data base can provide high flexibility and high extendibility.

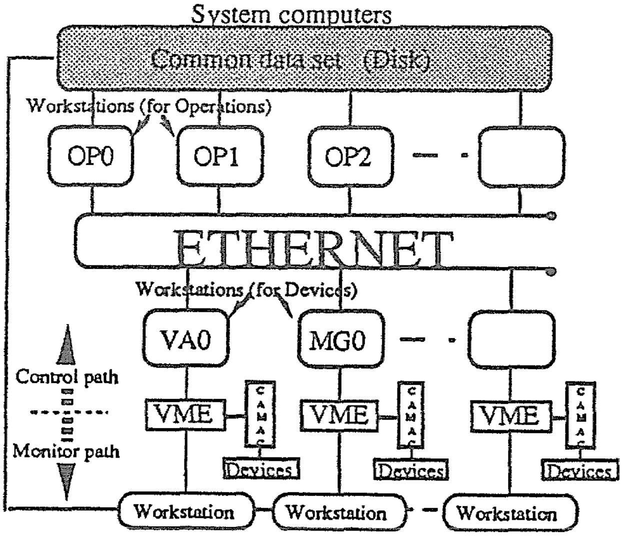


Fig 2 (B) : The possible new control system for TRISTAN using VME modules.

The discussion points are how to link these CPU's and how to handle the real time processes. The system architecture should be simple and the status of each CPU can be seen easily from operator consoles. From the point of this view, the number of hierarchy in the depth direction of the multi-computer system has to be minimized.

The real time response

In order to get high performance in the reasonable price, Unix workstations and VME modules are good candidates as components of the system. Since a Unix workstation does not have real time response, we have to prepare the real time system using VME modules or special real time process computers that are not well standardized yet. Two examples of the real time system are described in Fig. 2 (A) and Fig. 2 (B). Fig 2 (B) system uses VME modules connected to the Unix workstation to proceed the real time task. All processes that need real time response should be taken care at the VME level, so that the communication between operational consoles and workstations for the hardware equipment control is not necessary to be a token ring which has a real time response.

On the other hand, Fig 2 (A) uses real time Unix or VMS workstations to control real time processes. Since real time Unix and VMS workstations are vendor dependent, the system that deals with real time processes has to be minimized for the future update of the system. And these real time computers must be linked with a token type network such as FDDI.

The software development

The environment of software development is one of the key of successful system. The language should be available in the standard workstation. And programs have to be written easily. NODAL and Object oriented programing are good candidates. We should think about the usage of the commercial software package to save the time.

4.The modularization of the system

When we rejuvenate the control system, the key of the rejuvenation is modularization, because they have to be replaced step by step. In this section, the possible modularization with respect to both the hardware and the software is described.

The modularization of the hardware system

There are some possibilities in the hardware modularization.

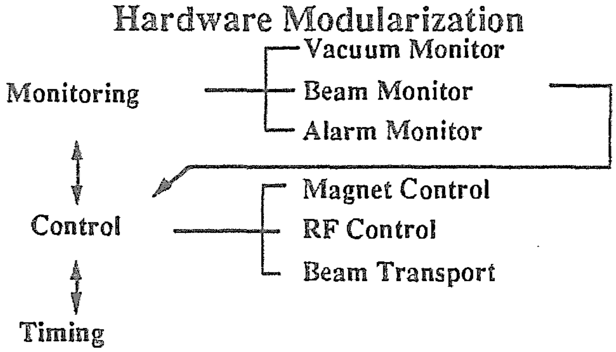


Fig.3 Separation of monitor path and control path

The monitor path can be separated from the path for device control. (Fig 3) The real time response is necessary for only the path of the device control. As same as the current system, the accumulator ring and the main ring can be modularized. It is also possible to modularize the system due to the function of each system, such as vacuum control, RF control, magnet control.

The modularization of the software system

The modularization of software is very important to rejuvenate the system and also important for the object oriented programing. The discussion in this context based on the four levels of control process.

Level 0 : The interface against operator. It must be possible to generalize the action of operators, such as 'select menu', 'input number', 'display data'.

Level 1 : The programs of this level should be written as the words of accelerator operation. For example, 'Measure Closed Orbit Distortion', 'Accelerate the particles','Change tune'.

Level 2 : The programs of this level are discussed as the protocol of the magnet or beam instrumentation operation, etc [5]. The objects of this level are the equipment of an accelerator such as a magnet, a vacuum pump, etc. And there should not depend on the sort of hardware interface, i.e. CAMAC or VME.

Level 3 : For a while, CAMAC has to be used for device control. All CAMAC actions should be defined in this level. When we want to replace some of the CAMAC modules to VME or other hardware modules, the software modification should be done in this level only. The codes should be put in a object library for each hardware individually so that it can be possible to update each hardware control system independently.

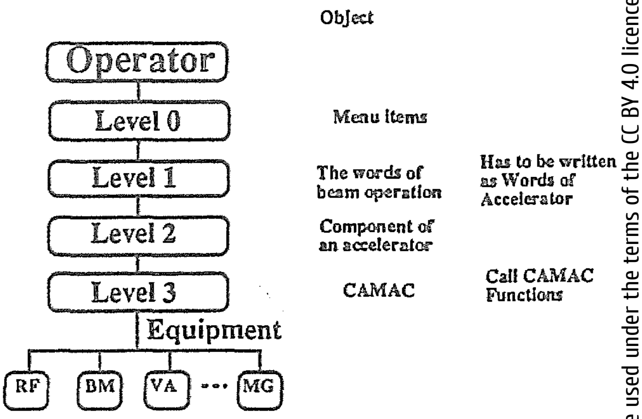


Fig 4 : The software modularization in the object oriented programing for the accelerator control

In each level, the object can be defined easily in the object oriented programing. Since they are well modularized, if CAMAC is replaced by other front end modules, such as VME, the software rejuvenation can be done easily. There must be no effect in level 0 - 2. And the part of the programs in level 0 - 2 can be shared in the different accelerators, for example B-factory and the main ring synchrotron radiation project.

5.The transition to the new generation control system

We have just started rejuvenating our control system. Three Unix workstations are installed and we start including them for our accelerator control. One workstation is going to be used to communicate with the linac control system. Since the current control system can not control linac parameters, such as the beam energy, the beam position of the injection line, it will be used to monitor and control some of the linear accelerator parameters and at the same time it sends some informations of TRISTAN ring to their control system. The other workstation will be used to do simulations for an accelerator control. The current control system has used PETROK that is the program for the computation of the closed orbit distortion correction [6]. And recently associating with the implementation of the superconducting Q-magnet, the SAD (Strategic Accelerator Design) which is the set of programs for simulation and optics matching, started to be used very frequently for the TRISTAN accelerator control [7]. The SAD programs are developed for accelerator design in KEK accelerator department. All programs are developed in the main frame: Hitachi HITAC. But the recent drastic improvement of the calculation speed of workstation make it possible to run the simulation in the private workstation. Since the current control system does not support standard network, the project to transfer the data from the current control system to the new installed workstation through CAMAC has just been started.

Because the current process minicomputers are 16 bit machine, the lack of memory causes some troubles.

The graphics has critical problems, so that it will be replaced by the graphics on workstations. In order to remove the current graphic system, the current software codes also must be available in the new workstations until all graphic programs are rewritten. The current graphic system "HIDIC 80-RS232-Graphics controller-CRT" , will be replaced by "HIDIC 80-RS232-Xwindow in the workstation"

The new functions for the accelerator control, such as the real time tune monitor, is going to be constructed with the graphics in the workstations

The possible system in the transition stages

There are several possibilities of the system rejuvenation. One possibility is to install another crate controller into each CAMAC crate so that two independent system can exist until the new system is completed. In any case, the replacement must be done step by step. The biggest change will be network support, unfortunatly at this point, the step by step replacement is impossible. Since the number of programs for man-machine interface are much more than that of hardware control, one possibility is to replace operational consoles with new network system in the early statge, and connect the new system and the old system through the gate way. For a while, both old and new operational consoles must be available. Then hardware process computers can be replaced one by one.

There are several options of this replacement. For example, 1: replace monitoring path such as a vacuum monitoring system, an alarm system first, then replace control path such as a RF control system, a magnet control system.

Actually there have already existed the independent RF control system composed of VAX workstations for taking care of aging of RF cavities. The system can be available for the system replacement.

Another method is 2: replace process computers for accumulator ring operation first, then replace that of main ring operation.

These hardware replacement has to be done together with the software emulations. But the software replacement is more complicated. One of the approach is to try to borrow NODAL program coded in C language from CERN. But even in such a case, all data module subroutines coded in PCL language have to be rewritten in C.

6.Conclusions

The rejuvenation of the TRISTAN control system has been just started to satisfy the increased requirements from complex operation of the accelerator. The Unix workstations and VME's are good candidates as a component of the new system, and some of them have already been added in the current system. The several R&D 's are underdevelopment in both hardware and software. We are going to choose one of the best solutions, but the basic ideas are

1: Use standards and make the open system. 2: Modularize the system and replace them step by step. 3:Use a commercial products to save a time and concentrate on the application program development.

References

[1] S.Kurokawa et al., Nucl. Instr. and Meth. A247 (1986) 29
[2] H.Ikeda et. al., IEEE Trans. Nucl. Sci. NS-28 No.3 (1981) 2359
[3] S.Kamada et. al., KEK Preprint 91-76 (1991)
[4] E.Kikutani et. al., The 6th Symp. on Accelerator Science and Technology. [1987] 236
[5] R.Bailey et al., Nucl. Instr. and Meth. A293 (1990) 332
[6] PETROK is KEK version of PETROS which is developed at DESY.
[7] K.Oide, Nucl. Instr. and Meth. A276 (1989) 427

Upgrade Plan for the Control System of the KEK e⁻/e⁺ Linac

K. Furukawa, N. Kamikubota, K. Nakahara and I. Abe
National Laboratory for High Energy Physics (KEK)
Oho, Tsukuba, Ibaraki, 305 Japan

Abstract

The KEK 2.5-GeV linac has been operating since 1982. However, in order to maintain reliable operation, the control system should be upgraded within a few years. We plan to replace the minicomputers and the main network connecting them. Thus, the architecture of the control software will also be revised. In the new system we should adopt software and hardware standards.

In the next control system we will employ the TCP/IP (DARPA) protocol suite for the main network and Unix workstations to replace the minicomputers. For connections to the local controllers, VME bus (IEEE 1014) will be utilized.

I. INTRODUCTION

The KEK 2.5-GeV linac provides electron and positron beams for both the TRISTAN collider and the Photon Factory storage ring. The linac control system was designed in 1979 and has been successfully operating since its commissioning in 1982. This old system is based on a distributed processor network, comprising eight minicomputers and hundreds of microcomputers, and on a control message exchange. Detailed descriptions have been given elsewhere [1] [2].

We have recently had many requests for increased functionality of the control system not included in the original design. However system resources were inadequate to implement the desired functionality. Furthermore, the minicomputers and the associated main network used in the old system will become unsupported by the computer company in a couple of years. We had thus decided to replace the main part of the control system and have carried out studies regarding the new system.

As a result, we have decided to adapt international and de facto standards as much as possible. The recent advances in electronic technology enable us to use common standards among various fundamental components. We have employed these standards in the new network system and computers. For the main network the TCP/IP (DARPA) protocol suite on the Ethernet (IEEE 802.3) media is utilized. Unix operating systems on a group of workstations and VME-bus-based (IEEE 1014) computers were chosen for the main console stations and subcontrol stations, respectively.

We describe below the design, current status and future of the new linac control system.

II. SYSTEM COMPONENTS

A modern control system must interlink many kinds of objects and facilities. We may link it with advanced technology, such as A.I. systems [3] or CASE tools. We may also link to the control systems of other accelerators or utilities, although some access limitation mechanism is needed. We may further link the new control system at the next upgrade time. Since the accelerators for research projects are always improving, we must combine many kinds of components to the control system. Thus we must make the control system as flexible as possible.

In order to fulfill the requests we cannot rely on only one hardware and software vender. We had better utilize international and de facto standards. If we have much capacity to develop control components, we may define our own standards. However, this would lead some difficulties in future, as we have already experienced. We thus want to utilize currently available standards.

The lifetime of an accelerator control system is relatively short, since electronics technology makes rapid progress and demands for control systems are changeable. We must thus always worry about future changes and upgrades of the control system. Since new technology and future standards are very likely to support existing standards, employing standards is promising.

A desire for standards has grown among both users and vendors recently. If we adopt standards, (a) since there will exist less ambiguity in the rules, we can develop reliable technology based on it, (b) since the required components may be obtained commercially, the development period may be substantially reduced, (c) since the standards in other fields may support each other, it may be easier to construct the entire system. These facts are especially important in a small system, such as in our case where manpower is limited.

A. TCP/IP Network.

Since equipment used for a accelerator is spread over a long distance, we must use some kind of computer networks between controllers. Since our linac is a pulsed high-power RF machine, in the old system we attached importance to noise elimination and have used proprietary fiber optic networks (Loop-1). Since their hardware and software are dedicated to the old system, we cannot utilize them for any other configurations.

Thus, we had to think about introducing a more standardized network, which can be adopted to various configurations. In order to be independent from suppliers TCP/IP protocol suite (DARPA) is most appropriate. We have decided to use TCP/IP, even on VMS operating systems where the DECnet protocol is available.

We have developed programs which test the basic communication capabilities of TCP/IP, which proved capable of running on more than 15 different computer/operating system architectures without any modifications to the source codes. Performance measurements were also carried out with these programs, and a part of the results is shown in [4]. The round-trip response time of the typical workstations over the Ethernet is about a millisecond, which is appropriate for a control network.

For the network media we utilize Ethernet (IEEE 802.3) since its components are commercially available and cheap. We know that the CSMA/CD mechanism of Ethernet is less effective, compared with token passing. However, we can make use of it up to 5 M bit/s, which is adequate for our current needs. It is possible to replace the main network part in the future, without any modifications to our software, with FDDI network system, which provides a band width of 100 M bit/s and token ring network access mechanism.

B. Unix Operating System

Unix is not presently a real-time computer operating system. However, except for full priority scheduling, most real-time features have already been implemented into many Unix operating systems. At least the current Unix systems can have a better real-time response than the real-time computers existing 10 years ago. Synchronization between processes can be possible in a millisecond if the processes are locked on memory; it took about 10 milliseconds to switch between processes on the old minicomputers. It should be noted that the POSIX (IEEE P1003) for Unix standardization tries to include real-time capabilities.

The system call incompatibility between Unix systems from many venders, especially between BSD-based and System-V-based Unix systems, could cause some troubles. However, most of our software can be easily ported, since we tried not to use any unusual system calls. We currently develop software mostly on DecStation 5000's in the control system and SparcStation 1 on the laboratory network. We also use the Mitsubishi's MX3000II for the gateway between the old and new network systems, as described below.

C. VME Front End

In the old system one subcontrol station comprises a minicomputer and a CAMAC crate. Since we have only a few kinds of CAMAC modules and the VME bus standard (IEEE 1014) is widely accepted, we will replace the subcontrol stations with VME-bus-based systems.

The OS9 operating systems and MC 68030 processors drive the VME systems. We have already finished examining the TCP/IP communication capability. We made the same network software to run on Unix and VME-based systems. The results satisfied our purpose.

In order to communicate with local device controllers we continue to utilize fiber optic in-house local networks (Loop-2,3) for some time, since there exists about 300 microcomputer-based device controllers [5]. The VME modules for the Loop-2,3 network system and its associated software are under development. Newly designed device controllers may use VME systems and communicate directly to the main network.

III. ARCHITECTURE

A. General Network Service

We are using some general network services, which includes: (a) time synchronization, (b) file sharing, (c) printer sharing, (d) computer and network status monitoring, (e) relational database access, (f) logging message distribution, and (g) name services. These mechanisms are achieved without any effort by using standard network protocols.

Currently, ntp (network time protocol) is employed on the Unix operating systems and VMS systems for time synchronization [6]. Modifications to the source codes were needed on the System-V-based Unix systems. With ntp, the time difference between computers is regulated using a statistical method; under normal operation it remains at about several milliseconds. This is very important under a distributed environment, where common resources are shared between many processes on the network, in order to keep consistent information.

The NFS service provides a homogeneous disk file system access over the network. The components of operating systems are shared as well as the control system databases and system development environments. Recently, computer equipment has become very reliable. However, hard disks can easily fail and data cannot be saved after crashing. It is thus important to reduce the number of hard disks as well as to save the contents frequently. We intend to run most VME-based machines and some workstations using NFS without local disks.

B. Message Exchange Controls

Since the old system runs based on control message exchange [2], in the new control system it must be supported in order to keep the old software running. The old control messages depend on the physical structure of the network connection and the characteristics of the associated equipment or service. In order to adopt control messages to the new control system we have also developed a suite of new control messages with more realistic symbolic messages and object-oriented concepts [4]. We have already developed a message-conversion process between old and new formats.

C. Servers

In the old system although many processes are loaded onto the main console station., the number of processes running on the station is limited. It is thus difficult to expand the control system functionality. On the new system we should make the control software independent of the number of control workstations.

The response time measurement mentioned above suggests that the processes on a main console station can be distributed among many workstations, since the response time between the processes on the new network system is much faster than that on the same old main station.

In order to distribute the control processes, we should develop several processing servers which would function as an operating system on the control network. These servers should include: (a) a static database server, which would provide characteristics of equipment and services, and (b) a dynamic database server, which would distribute live accelerator status. These services need some caching mechanism on each computer. We should also provide: (c) a locking server, which could carry out resource management, (d) an alarm server, and (e) a communication server, which could manage communication to other accelerators or facilities. With these server processes the network system would act as one computer.

D. Surveillance and Diagnosis

We have realized that under normal accelerator operation the status and fault surveillance system is important for maintaining stable operation [7]. The purpose of the control system is to operate the objects based on some physical models. These models should be performed in the control processes. However, it is always possible that the equipment or software may fail. Surveillance processes should detect any failure and alarm the operators. The database information servers and alarm servers described above would help these processes. Preferably, fault recovery should follow fault detection with a diagnostic system.

Currently, most surveillance tasks depends on human operators. We thus need some simple method to describe the conditions. Otherwise, we must put complicated surveillance description into the software. Object-oriented software techniques will aid the in development to some extent. We also intend to utilize rule-based real-time expert systems.

E. Remote Procedure Calls

Although we are now developing control software based on message exchanges, we are also considering using the mechanism of remote procedure calls (RPC) in order to improve the efficiency of the system and application software production. We implemented the NC/RPC system of CERN's [8] and RPC of Sun Micro System's into DecStations and a SparcStation, which gives good results. With some improvements for software development environment and asynchronous RPC, we will use the RPC in near future.

IV. SCHEDULE

A. Condition

The operation of the accelerator doesn't allow us long shutdown periods of the control system. We must therefore replace the system components gradually.

The original system comprised 6 components as already described: (a) two minicomputers as the main console stations, (b) six minicomputers as the subcontrol stations, (c) a fiber-optic proprietary network between minicomputers, (d) hundreds of microcomputers for local device controllers, (e) about twenty fiber-optic in-house local networks, and (f) an operator's console subsystem of personal computers [9]. We will replace the first three components during the first stage. Then others will be improved.

B. Gateway Between Old and New Systems

Since it's risky to completely replace the system components all at once, we have built a control message and status information gateway between the old and new systems [2]. The gateway comprises software on the old and new computers which transfer information through parallel interconnections. Messages can be transmitted from the new network system to the old system, and vice versa, without worrying about gateway software.

Several application programs, such as equipment status surveillance which was limited in the old system, work fine over the gateway. The message-transfer scheme was also a basic concept in the old system; thus many old applications can be moved to the new system.

We are currently developing software for the new system using this gateway. Although the response time on the old network side is large, the new software should well simulate the new system

We have no control computer networks to the TRISTAN control system. However, we need to communicate with each other for better operation. These systems will be connected through the TCP/IP network during this year. Using TCP/IP protocols, message-packet routing, as well as its limitation, is possible without using any additional software.

C. Main Control Workstations, Subcontrol Stations and Main Network

Since the original main network is a proprietary for old minicomputers, we cannot install these three separately. However it is not too difficult since the gateway system described above is already installed. We will first change the route of messages to the Loop-3 networks into new system. After that, the route to the Loop-2 networks will be switched. The physical connection and message flow will switch as shown in figure 1. The replacement is scheduled to take place over a period of two years.

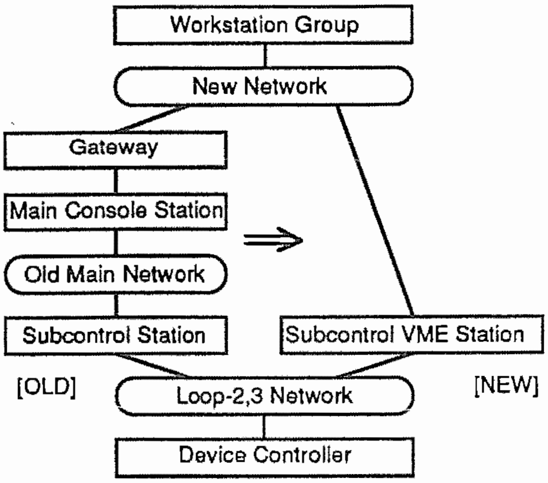


Figure 1. Transition of the system connection and the control message flow from the old system to the new system.

Later, the network traffic will increase since we share it also for software development. We will thus separate the network into two parts: a network for subcontrol stations and a network for main control stations. It is also possible to replace the latter with an FDDI network, which is more efficient, employing token ring network access mechanism. The total system is shown in figure 2.

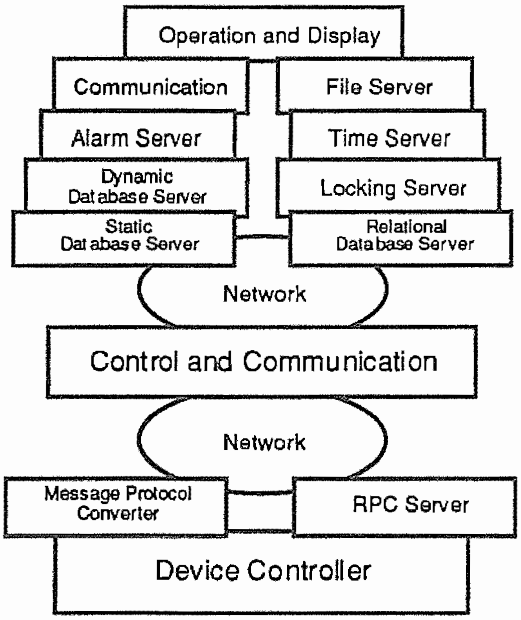


Figure 2. Logical structure of the new linac control system.

D. Further Improvements

Although we continue to use the old microprocessor-based local device controllers, for the new kinds of equipment we will utilize VME-based local controllers. During the next year we will obtain several pieces of new accelerator equipment, that may be the first case to have new local controllers which are connected directly to the main network.

For the operator's console, a personal-computer-based system which enables frequent modifications, is utilized. However, the connection to its gateway personal computer is a serial link, and the new functionality on the new control system cannot be utilized from that system. We are thus presently developing some console functions on X-Window. The application development environment has improved much using various window toolkits and widgets. We will utilize Motif widgets for application programs. Some simple applications has been developed and an X display will be installed into the operator's console room during the next year.

V. CONCLUSION

An upgrade scheme of the linac control system was investigated and we have obtained good results so far. Using international and de facto standards, the upgrade has become much simple and we need not rely on any single company. This is important in small accelerator control systems, as in our case.

VI. ACKNOWLEDGEMENT

The authors wish to thank Professor A. Asami for his support during this work. They also would like to thank the staff and operators of the linac for their valuable discussions during the system design and development. One of the authors (K.F) is grateful to Dr. K. Kostro who allowed him to use the NC/RPC code of CERN.

VII. REFERENCES

[1] K. Nakahara, I. Abe, A. Enomoto, Y. Otake and T. Urano, Nucl. Instr. and Meth. **A247** (1986) 153.
[2] K. Furukawa, N. Kamikubota, K. Nakahara and I. Abe, Nucl. Instr. and Meth. **A293** (1990) 16.
[3] I. Abe,, M. Kitamura and K. Nakahara, "Network communication libraries for the next control system of the KEK e^-/e^+ linac", this conference.
[4] N. Kamikubota, K. Furukawa, K. Nakahara and I. Abe, "A diagnostic expert system in PF linac control", this conference.
[5] For example, T. Shidara, R. P. Bissonnette, K. Nakahara and S. Asami, Nucl. Instr. and Meth. **A245** (1986) 500 .
[6] D. L. Mills, DARPA Network Working Group Report RFC-1129 (1989).
[7] K. Furukawa, N. Kamikubota, T. Shidara, K. Nakahara and G. Horikoshi, Part. Accel., **29** (1990) 245.
[8] P. S. Anderssen, V. Frammery and R. Wilcke, Nucl. Instr. and Meth. **A293** (1990) 225.
K. Kostro, private communication.
[9] K. Nakahara, I. Abe, K. Furukawa and N. Kamikubota, Nucl. Instr. and Meth. **A293** (1990) 446.

The New Control System for TARN-2

S. Watanabe, M.Yoshizawa, J.Yoshizawa and T.Katayama
Institute for Nuclear Study, University of Tokyo, Tanashi, Tokyo 188, Japan

K.Aoki and K.Ohnishi
Sumitomo Heavy Industry, Tanashi, Tokyo 188, Japan

The new control system for the cooler-synchrotron, TARN-2, is described. The new control system consists of OPU's (work stations) and EXU (control computer) linked with the local area network. The text message is used to transfer the control commands and their results. The control program CSA90 at EXU decodes the text message and executes it with the aid of the interface and periodic control subroutines. Both subroutines use common sharable image composed of the status, values, parameters and so on. The CAMAC, GPIB and RS232C are standard interface at EXU.

Introduction

A heavy-ion synchrotron-cooler ring, TARN2 has been constructed at the Institute for Nuclear Study, University of Tokyo. It has a maximum magnetic rigidity of 6.2 Tm, corresponding to 1.1 GeV for protons and 0.37 GeV/u for ions of charge-to-mass ratio 1/2. Now the injector is a sector-focusing cyclotron with K=68. The aims of TARN2 are to study the acceleration, cooling and extraction of heavy ions and so on. Recently study of the weak beam monitoring and experiment of atomic physics is being continued. So the TARN2 control system must gives reliable control functions and monitoring required to direct the up-to-date complex operations associated with on site experiment.

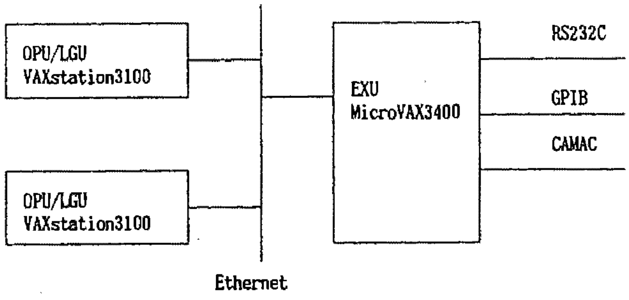


Fig.1 The new computer control system of TARN2

The present control system consists of the serial CAMAC, GPIB and microcomputers[1]. The serial CAMAC system covers static control system such as beam transport[2], injection, electron cooling[3] and a beam extraction system. On the other hand, dynamic control of both the RF acceleration and ring magnet system[4], is also performed with the dedicated microcomputers with external memory modules[5]. These systems well regulate TARN2 and now used without some troubles.

In 1990, basic design of the new computer control system was started and was authorized to combine the present I/O controllers. The present paper describes the new control system of TARN2 during the development phase.

Basic Architecture

In the new control system, the actual device control and man-machine communication is made by the separate units. The new system comprises OPU's (workstations) and EXU (control computer) linked with the local area network. The new computer control system of TARN2 is shown in Fig. 1.

The several kinds of workstations and PC's are used as OPU's. One is a VAXstation 3100. Another one is a microcomputer NEC-PC9801 with Ethernet board and DECnet-DOS. The uVAX 3400 is used as EXU and CAMAC, GPIB and RS232c are also available in the EXU.

The OPU and EXU execute task to task communications between them. Before the message transmission, OPU is linked with EXU. Then OPU and EXU transfer the text message to each other. The format of text message is formalized as shown in Fig.2.

At EXU, the received text message is stored into a buffer area by the interface process of control program CSA90[6]. Subsequently the buffered text message is taken by interface subroutine and then executed by a periodic subroutine after decoding the text message. Both subroutines use a common sharable image composed of the status, values, parameters and so on. Both subroutines are registered in the EXU and selectively called by the OPU using the subroutine number. The text message consists of this interface subroutine number and a command message associated with a control procedure of the target devices. So the central job at OPU is to generate the text message according to the associated control procedures. The interval of a periodic subroutine can be controlled by the associated control program CSA90. The minimum interval time is 50 msec and can be increased by 50 msec integrals.

The communication network 'INS-Ethernet' is a standard network in our institute, whereas other network system is used in our institute. The new control system is linked with the local area network which is in the TARN2 and cyclotron areas. Our network is also linked with 'INS-Ethernet'. On the other hand, the large scale computer system INS-M780 is also available through the associated local area network if we need any large computing power. In this case, TCP/IP protocol is used and terminal server with LAT-TCP/IP protocol is available in the network system.

The log-in by the off-site users will be expected through the network. To reject such a problem, physical cut off of the network will be expected during the actual operation phase, but general protection method is also employed.

function	input	output
get device information number	0:D<devnam>	0:n
get interface subroutine number	0:R<subrnam>	0:n
get periodic control subroutine number	0:K<subrnam>	0:n
get device information name	0:n	0:<devnam>
examine information	0:@Cn	0:<string>
	0:@In(,m)	0:x
	0:@En(,m)	0:x
modify information	0:@Cn=<string>	0:
	0:@In(,m)=x	0:
	0:@En(,m)=x	0:

() = optional
i = subroutine number
n = item number of device information
m = element number of array
x = data

Fig.2 Format of Text Message of OPU/EXU

Operator Console

The distributed operator consoles provide many kinds of operating functions. Up to 10 operator consoles could be joined with EXU to execute task to task communications. The main aim of task to task communication is to transfer the text message as described before. On the other hand, operator consoles as workstations provides us with the highest quality of visualization, man-machine communication and calculation speed. We have been considering several types of man-machine communications such as a touch panel, rotary encoder, mouse and so on.

The NEC-PC9801 with the resistive touch panel sheet and rotary encoder is chosen. The mouse system is also used in this OPU. The ON/OFF control and value setting are the main job of this system. The operating system of MS-DOS 3.3B and MS-C, MASM and turbo-C have been used to develop the application control program. DECnet-DOS is used in this system as the communication interface to EXU.

The VAXstation 3100 is used as OPU with VAX resources. The beam simulation program is available in this system. For instance, COD correction can be performed by the combination of OPU and EXU. The EXU sends measurement result of a beam orbit to OPU and receives back the command to change the correction current. The change of current is carried out by EXU.

Monitoring of the operating condition is an important facility. To discover the operating status, EXU periodically collects the operating status through the CAMAC system. The collected data is saved into a file and periodically refreshed by a timer process. The OPU assigned as a warning monitor refers the file in any time and executes appropriate fail-safe procedures.

Control Computer

The control program, CSA90, has been developed to execute the interface and periodic control process of EXU. The logical image of CSA90 is shown in Fig. 3.

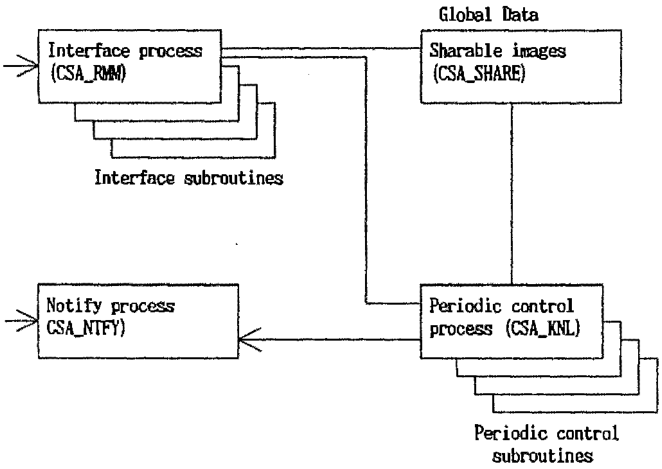


Fig.3 Logical image of control program CSA90

The CSA90 provides us with a flexible and expandable control system. This program has a basic program, device control programs, configuration definition files and a data generator. The device control programs are a set of subroutines and have no direct relation to each other. The basic program executes those subroutines according to a configuration program and requests from OPU. To make the configuration definition file easily, a data generation program is also provided.

By using CSA90, addition of new functions or modification of the control procedures are easily made by addition or modification of subroutines. Then the control sequence or configuration of the control system could be changed easily in according to the purpose of accelerator studies and so on.

The I/O interfaces are comprised of CAMAC, GPIB and RS232c. Presently the main CAMAC crate is alternatively controlled by either the Kinetic 3922 and EXU or Kinetic 3920 and an old existing microcomputers. The serial CAMAC driver housed in the main CAMAC crate is used commonly by either EXU or existing microcomputers through the above mentioned crate controller. Several kinds of local control devices with GPIB are located near the RF acceleration system, electron cooling and elsewhere. They are controlled by EXU through the fiber cable. The CAMAC control system also supports GPIB through the local CAMAC-GPIB converter.

Local intelligent Control Devices

The distributed control devices with an intelligent function have been used in the TARN2 control system. For instance, pattern generator to regulate the magnetic field of the ring has been used in the synchrotron magnet power supply system. Present pattern generator is composed of the microcomputers and is directory controlled by the terminal at the control room. To include such an individual intelligent controllers into the network system, we have been considered an up-grade of existing intelligent one to new one.

Though the add-on board of Ethernet system can be used in the present intelligent local control system, it is difficult to construct the network communication system because it is depend on the operating system of the central processor of local control system. For more any unknown conditions, we have already considered that distributed local intelligent devices should be tied with the local area network. For instance, above mentioned pattern generators will be changed to a VME system with pattern memory system. The local CAMAC computer housed in the CAMAC crate is also considered as the candidate of distributed intelligent local control device. We have already been developed 8bit CAMAC module computer with a function of pattern signal generator, parallel bit signal input and output for any purpose. In this system, control program is stored as the ROM image and the RAM image. In Fig.4 the block diagram of proposed 32 bit CAMAC board computer is shown.

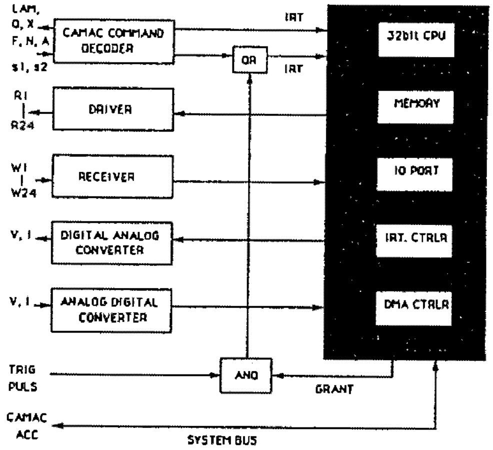


Fig.4 Proposed 32bit CAMAC board computer

Conversion from present system to new one

The present microcomputer systems is written using an interpreted language such as INSBASIC with CP/M-86. To continue this approach with the present I/O control and the new control system, especially for OPU, a program converter has been developed from INSBASIC to QuickBASIC, which is used as the programming language for OPU. So the new control system will provide us with the user functions and subroutines associated with the present INSBASIC. Thus transparent programming can be performed in the new control system, especially for PC9801 as OPU.

For the FORTRAN system, such as the minicomputer U-400 or HP1000, the programs developed for CAMAC system will be implemented on the OPU and EXU. For instance, an existing program indicating the beam envelope of transport line will be reconstructed with MAGIC code in VAXstation 3100 and CAMAC handler in uvax 3400.

References

[1] S.Watanabe, The Computer Control System of TARN-II, Nuclear Instruments and Methods in Physics Research A293(1990)50-53.

[2] F.Soga, K.Noda, K.Chida, T.Hattori, T.Honma, T.Katayama, A.Mizobuchi, M.Nakai, A.Noda, F.Sakuta, M.Sekiguchi, N.Ueda, S.Watanabe and M.Yoshizawa, Beam Transport and Injection System from the SF Cyclotron to the TARN-II Synchrotron Cooler Ring, INS-T-494, February 1990.

[3] T.Tanabe, K.Chida, T.Honma, T.Hattori, M.Kanazawa, A.Mizobuchi, M.Nakai, A.Noda, K.Noda, M.Sekiguchi, F.Soga, N.Ueda, S.Watanabe, T.Watanabe, M.Yoshizawa and A.Ando, Initial Electron Cooling Studies in the TARN II, Proc. of the 7th Symposium on Accelerator Science and Technology, Dec. 1989, Osaka, Japan, p201.

[4] S.Watanabe, T.Kohno, S.Yamada and T.Katayama, Power Supply Control System for TARN-II Main Ring Magnet System, Proc. of the 6th Symposium on Accelerator Science and Technology, Oct. 1987, Tokyo, Japan, p242.

[5] S.Watanabe, A.Noda, T.Watanabe and T.Kohno, Computer Control of TARN-II Main Ring Power Supply, Proc. of the 7th Symposium on Accelerator Science and Technology, Dec.1989, Osaka, Japan, p249.

[6] K. Aoki, S. Watanabe, The Design of the New Control Program (CSA90) for the TARN2, INS-T-505, 1991.

A New Architecture for Fermilab's Cryogenic Control System

J.Smolucha, A.Franck, K.Seino and S.Lackey

Fermi National Accelerator Laboratory*
Box 500, Batavia, Illinois, 60510

Abstract

In order to achieve design energy in the Tevatron, the magnet system will be operated at lower temperatures.[1] The increased requirements of operating the Tevatron at lower temperatures necessitated a major upgrade to the both the hardware and software components of the cryogenic control system. The new architecture is based on a distributed topology which couples Fermilab designed I/O subsystems to high performance, 80386 execution processors via a variety of networks including: Arcnet, iPSB, and token ring.

Introduction

The addition of "cold compressors" to the Tevatron's satellite refrigeration system, as well as the desire to dynamically balance the site's distributed compressor load, introduced additional performance requirements on the cryogenic control system. The required signal processing capabilities basically doubled, raising the number of I/O terminations to approximately 2,000 per satellite.

To effect global optimization the system's software also required significant enhancements. These included: support for global communications between refrigeration processors, an improved user interface to the finite state machines, both code and parameter down-loading capabilities, and substantially improved diagnostic support for the system's hardware.

Together, the performance criteria exceeded the capabilities of the original, Z80 based control system[2]. Therefore, a complete upgrade was warranted.

The new architecture

To achieve the desired performance with a minimum of hardware, we adopted a distributed architecture centered around Intel's 32 bit, 80386 microprocessor. Multibus II was selected as the hosting platform. We prefer it not only for the architectural advantages it offers, but also because of the existing support services which have already been developed for our front end computers[3]. The substantial amount of hardware and software that we were able to inherit allowed us to concentrate the majority of our efforts on developing smaller, more efficient I/O subsystems.

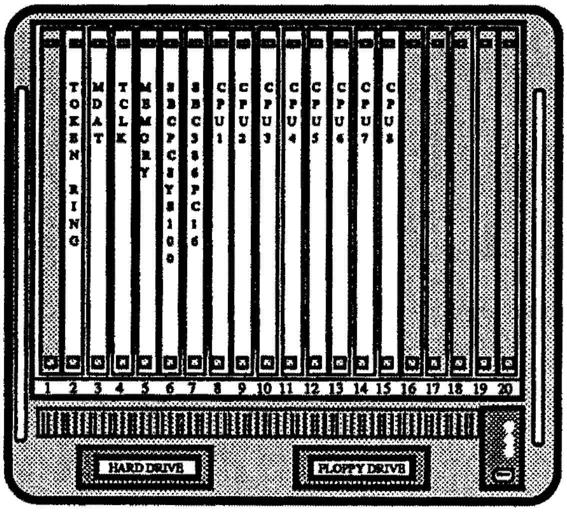
We partitioned the control system into six sectors following the physical distribution of satellite compressors around the Tevatron. Each sector consists of four satellite refrigerators and their associated compressor. The sectors are connected by a token ring network to effect global communications.

Each satellite unit is controlled by an Intel, iSBC 386/120 single board computer. The five CPU modules reside in a common Multibus II chassis which is located at the compressor installation. The loosely coupled architecture of Multibus II is ideally suited to this application. It supports protected, independent execution for each processor module, while simultaneously providing fair access to centralized system services. These services include: a Tevatron clock distribution processor, a "machine data" (MDAT) I/O module, global shared memory, a DOS based 386PC/AT for system initialization and diagnostics, and a token ring processor for global communications.

Inter-processor communications within the sector are accomplished over the parallel system bus (iPSB) which, in conjunction with the message passing protocol of Mutibus II, functions as a 40 megabyte per second local area network. The crate configuration is illustrated in Figure 1.

---

* Operated by the Universities Research Association Inc., under contract with the U.S. Dept. of Energy.



MULTIBUS II CRATE

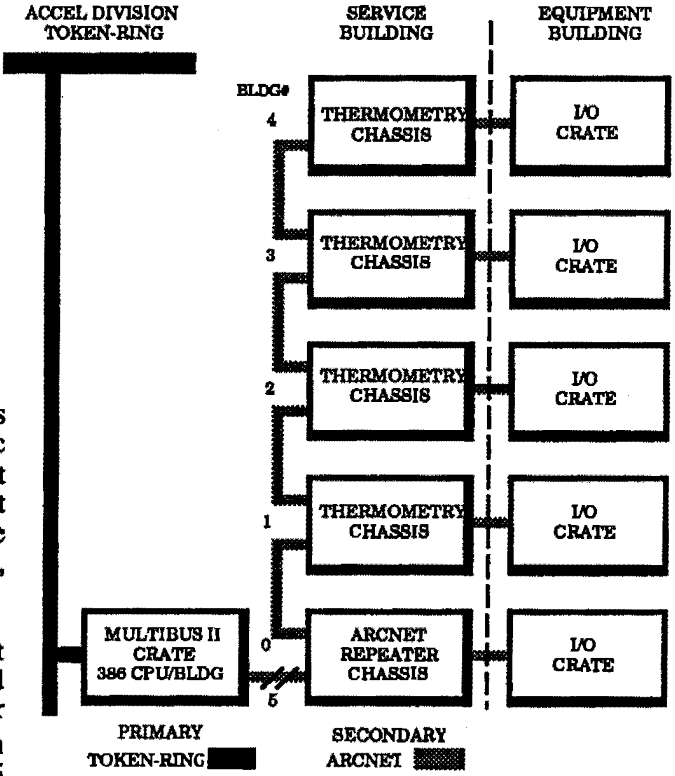
FIGURE 1

Subsystem communications

Control for each of the satellite refrigerators is further divided into two I/O subsystems: Cryogenic Thermometry and Device I/O. (Compressors do not require thermometry .) The subsystems are located at the source of their respective signals and are connected to the execution processors by Arcnet, which functions as a small area network.

Arcnet is a low cost, highly reliable network that supports up to 255 nodes and can be implemented over a variety of media including coax, twisted pair and fiber[4] . It utilizes a deterministic "modified token passing" access scheme with a data rate of 2.5 megabits/second and a packet size of 512 bytes. It has a very simple protocol and all of the network services are provided at the silicon level by a variety of commercially available controller chips. The controller interfaces to a system through a page of dual ported RAM, and a single command/status register. To initiate a network transmission the host processor simply loads a message packet into the RAM buffer and issues a "transmit request". Message packets consist of a source ID, a destination ID, the message byte count, and up to 508 bytes of user defined data. All messages are accompanied by a 16 bit CRC character which the receiving node uses to verify the integrity of the transmission. Additionally, all error free transmissions are positively acknowledged.

There are no hardware imposed restrictions on communications within a sector. Each node is permitted equal access to all the other nodes on the network, thereby allowing a variety of logical topologies. Currently, a Fermilab designed protocol layer called "Virtual I/O Bus" (VIOB)[5] , manages communications between the execution processors and the I/O subsystems. VIOB enables the subsystems to appear as extensions of the execution processor's local buses. The current topology for a single sector is illustrated in Figure 2.



COMMUNICATIONS TOPOLOGY

FIGURE 2

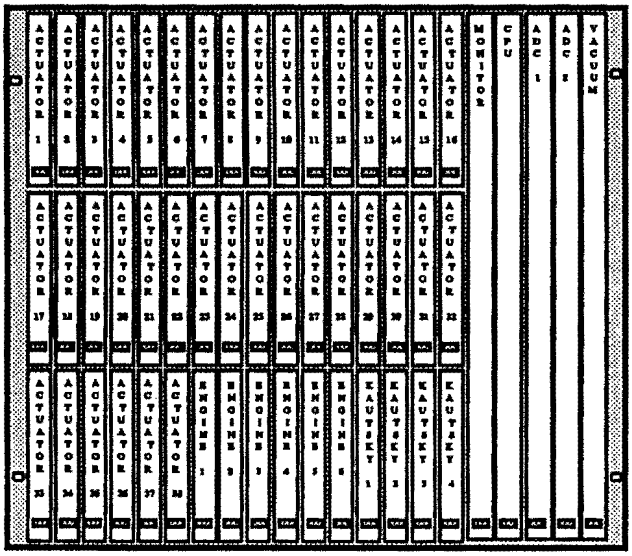
The I/O subsystems

The basis for each of our subsystems is a 16 Mhz Intel 80C186EB functioning as an I/O processor. The "EB" is a low power, CMOS version of Intel's 80186 embedded microprocessor. It is based on a modular CPU core that integrates most essential system services onto a single 80 pin package. The object code of the 186 is directly compatible with the "real mode" of the 80386 microprocessor. Hence, a subset of the tasks developed for the execution processors could, in the future, be ported to the I/O processors as a way to further improve software performance.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst.
ISBN: 978-3-95450-254-7
ISSN: 2226-0358
ICALEPCS1991, Tsukuba, Japan
JACoW Publishing
doi:10.18429/JACoW-ICALEPCS1991-S02SRU11

The Device I/O subsystem

The Device I/O subsystem includes support for 128 transducer inputs, 38 actuator valves, 32 helium relief valves, 12 vacuum gauges and 6 "wet" or "dry" expansion engines. The subsystem is housed in a Euro-chassis located within the cryogenic equipment building. The current configuration is illustrated in Figure 3.



EQUIPMENT BULDING I/O CRATE
FIGURE 3

Activity within the subsystem is orchestrated by a 9U size processor module which hosts the 80186EB. In addition to those system services provided by the 80186, the processor module also provides fast data logging facilities, Tevatron clock decoding, the Arcnet interface, and distribution of various clock signals.

The I/O processor interfaces to the rest of the system via three identical backplane buses referred to as the "F2" bus. The F2 bus was developed in house to overcome the difficulties associated with cabling an average number of process I/O signals to a 3U size Eurocard while simultaneously supporting a high performance 16 bit bus interface on a single 96 pin DIN connector. It provides 32 pins of user defined I/O at each slot along with addressing space for up to 512 sixteen bit memory mapped registers. The individual slots are mapped to the I/O processor's memory and are individually selected in a fashion similar to the CAMAC standard.

Transducer interfacing is accommodated by two, 9U size A/D converter modules. Each module supports 64 differential input channels with a 10 volt range. The analog front end consists of two stages of input multiplexing, an instrumentation amplifier, and a 12 bit, self calibrating ADC. To facilitate fast data logging all 64 channels are digitized within a one millisecond period. The results are stored in a double banked, dual ported RAM. The arrangement allows the I/O processor fast access to the digitized data with a minimum of arbitration conflicts.

Actuator Control cards provide a comprehensive interface to the Barber–Coleman electromechanical valve actuators used throughout the cryogenic system. Each 3U size card incorporates all of the functionality necessary to interface a single device including: 24 volt DC drive control, Linear Variable Differential Transformer instrumentation, and A/D conversion circuitry. A local front panel control mode is also provided.

Wet engines, dry engines and cold compressors[1] all interface to the control system via Engine Control cards.[6] Each 3U card provides momentary relay contacts to implement "start", "stop" and "reset" commands. Also included are two independent channels of analog I/O (one for engine speed and one spare), and 16 optically coupled status bits. A front panel display reflects the state of both command and status bits.

The variety of solenoid valves used throughout the cryogenic system are interfaced by Kautzky[7] Control cards. Also implemented in the 3U format, each card provides control and status readback for up to eight solenoids. A front panel display reflects both the actual state and the requested state of each valve.

Pirani (vacuum) gauges are interfaced by a 9U size Vacuum card. The card supports 12 transducer circuits. A free running, multiplexed A/D converter continuously scans the transducer voltages, storing the digitized results in a dual ported RAM buffer.

Miscellaneous status bits are monitored by a Digital Input card. The 3U size card supports 30 bits of optically coupled status with input voltages ranging from 5 to 24 volts D.C.

The thermometry subsystem

The thermometry subsystem supports 96 channels of pulsed current, resistance thermometry. Unlike the Device I/O subsystem, it is implemented as an application specific single board computer. It resides in a NEMA–12 enclosure mounted along the back wall of the Tevatron Service building.

The free running measurement scheme uses a precision current source (along with six stages of multiplexing) to deliver a 50 microsecond current pulse to each resistor module. The voltages developed by the resistors are scaled by a programmable gain instrumentation amplifier before being digitized by a 12 bit, self calibrating A/D converter. Here too, a dual ported RAM is arrangement is used to store the digitized results.

The subsystem also maintains a hard-wired 16 bit bi-directional data link with the Tevatron Quench Protection system (QPM). In the event of a magnet quench, the QPM provides the refrigeration control system with specific cell information to assist automatic recovery schemes. The refrigeration system provides the QPM with a permit allowing the magnets to be turned on.

Reliability and system diagnostics

One of the notable benefits of using the Multibus II architecture is the emphasis that the platform places on confidence testing and system diagnostics. Every module in the system supports a standardized set of "built in" self tests (BIST). During the initialization phase following a power–on reset, each module executes a subset of these routines to test its own functionality. The results are posted in "interconnect space" and are globally available to the rest of the system. Upon the coordinated completion of a reset, one module (in our case the PC16) assumes a temporary role as the systems' "boot master". The master can subsequently invoke, or download additional tests for each subsystem on an individual basis. This can be accomplished locally via an RS232 port or from a remote facility with the use of a modem.

Upon completion of confidence testing the operating system and refrigeration specific application software are down-loaded via the iPSB. At that time the execution processors are started and the PC16 relinquishes control of the system. The PC16 is now free to be utilized as a local information data base or console emulator by technical personnel. System schematics, diagnostic flowcharts and related maintenance histories can be displayed, as well as any other useful DOS based application.

Conclusion

The use of a hierarchical, networked architecture is ideal for this application. The higher cost, 32 bit microprocessors have been de-coupled from the cryogenic system, thereby protecting the investment from premature obsolescence. The distributed subsystems can be incrementally upgraded – or even replaced – with a minimum of impact to the rest of the system. Additionally, the maximum limit of 255 nodes per sector permits almost unlimited expansion of the system.

Acknowledgements

We wish to acknowledge the contributions of those individuals whose efforts were responsible for turning ideas into functioning components for this system, especially: Loren Anderson, Ryan Hagler, Joe Gomilar, Timothy Gierhart, Rich Klecka, Rich Koldenhoven, Robert Marquardt and Thomas Zuchnik.

---

[1] Fuerst,J.D. "Trial Operation of Cold Compressors in Fermilab Satellite Refrigerators", Advances in Cryogenic Engineering, Vol. 35B, Plenum Press, New York (1990) .

[2] Zagel,J. et al., "Tevatron Satellite Refrigeration Control Subsystem",IEEE Transactions on Nuclear Science, Vol NS28 , p 2153-2154.

[3] Glass,M., et al., "The Upgraded Tevatron Front End"' NIM A, 1990, Vol 293, p.87-92.

[4] Glass,Brett; "The Return of Arcnet", BYTE, February 1991, p119.

[5] Chapman,L. et al., "Microcomputer Software Architecture for Cryogenic Controls", (This conference).

[6] Rode,C.H., "Tevatron Cryogenic System", 12th International Conference on High Energy Accelerators, 1983. p 529-535.

[7] "The Kautzky Valve", Fermilab Reports, June, 1982.

Controls for the CERN Large Hadron Collider (LHC)

K.H. Kissler, F. Perriollat, M. Rabany and G. Shering
CERN, CH-1211 Geneva 23

Abstract

CERN's planned large superconducting collider project presents several new challenges to the Control System. These are discussed along with current thinking as to how they can be met. The high field superconducting magnets are subject to "persistent currents" which will require real time measurements and control using a mathematical model on a 2-10 second time interval. This may be realised using direct links, multiplexed using TDM, between the field equipment and central servers. Quench control and avoidance will make new demands on speed of response, reliability and surveillance. The integration of large quantities of industrially controlled equipment will be important. Much of the controls will be in common with LEP so a seamless integration of LHC and LEP controls will be sought. A very large amount of new high-tech equipment will have to be tested, assembled and installed in the LEP tunnel in a short time. The manpower and cost constraints will be much tighter than previously. New approaches will have to be found to solve many of these problems, with the additional constraint of integrating them into an existing framework.

I. LHC REQUIREMENTS

The Large Hadron Collider (LHC) is the major project planned by CERN[1], and will be its largest and most expensive ever. It will present control problems much greater than those experienced in earlier accelerators.

LHC is a superconducting twin beam hadron (proton initially) collider providing 7.7 TeV per beam at 10 Tesla bending field. The novel twin bore magnets in their cryostats will be installed in the same 27 kilometre tunnel as the LEP machine. The scale of the control problem can be gauged in part from the 1792 dipole and 392 quadrupole cryostats filling most of the circumference, in part from the number, about 2000, of insertion and corrector magnets and appropriate beam instrumentation. The **difficulty** of the control problem will come from the sensitivity of the superconducting magnets to quenching under beam loss from the 4725 bunches of 10^{11} protons at 400.8MHz, making 851mA. This problem is exacerbated by the time varying persistent currents and the need for strong collision insertions to achieve the targeted luminosity of over 10^{34}. These requirements will strain dynamic aperture and magnetic field control to the very limit.

As LHC will be built in the same tunnel as LEP, a lot of equipment and controls will be common to the two machines. A major objective will therefore be a seamless integration of LHC and LEP controls. This will not be easy, in part due to the much more difficult control problems of LHC, in part due to the wide separation in time between the construction of the two systems compared to the speed of evolution of controls technology. A challenge for LHC will therefore be to permit the use of the latest and cheapest controls technology in such a way that it integrates with existing technology, allows experience and algorithms to be maintained, and does not demand a difficult and costly upgrade of existing systems. Another objective to be borne in mind is the aim of having a single control centre for the whole of CERN on the LHC time scale.

II. MAGNETIC FIELD CONTROL

This will be the most difficult control problem for LHC. The magnetic field is determined not only by the voltages and currents from the 1400 power supplies, but also by "persistent currents" in the superconductor which vary with time depending on the history of the magnetic cycle. Fortunately HERA experience has shown these effects to be reproducible, hence eventually calculable and correctable. Corrections will be derived from magnet measurements during the construction and from on-line measurements from reference magnets. The final trimming will have to be done using single pilot bunches of first 10^9 then 10^{11} protons. After full beam injection, continuous feedback control will be required, especially immediately after injection and during beam squeezing.

A. Modelling Server

The solution envisaged is to use a modelling server which will re-calculate the power supply settings in real-time, using measurements from the reference magnets, beam measurements, past history of the magnetic cycle, and the magnet characteristic data-base. The update time will be between 2 and 10 seconds. Tests on an Apollo DN10'000 in the LEP control system indicate that the computational load will not be beyond the sort of on-line computer we can expect in the LHC time scale. Each power supply will have a microprocessor capable of interpolating the required voltages and currents between modelling server updates.

B. Fast Communications

A new communications system is being studied for LHC, in conjunction with SSC, in order to acquire the beam data and set the power supplies at the required rate[2]. This will use TDM communications technology and reflective memory computer cards to provide parallel transmission of data with no software protocol overhead. The power converters and beam monitors will use the latest digital signal processors and microprocessors, so no problem is envisaged at this level provided they can all operate in parallel.

C. Pilot Procedures.

It will be necessary to automate the procedures of machine preparation, pilot injection tests, and full beam injection and acceleration, in part to keep the persistent current effects under control, in part to avoid human error. Much work of this kind was done for the Antiproton Accumulator after initial commissioning, but for LHC this will have to be done beforehand, so placing an early load on the applications programming effort. Of course the modelling programs in the servers will also have to be ready and tested before injection tests can start. High quality work in these areas will greatly reduce the time needed to get the machine into proper working order.

III. QUENCH PROTECTION

The first line of quench protection will be assured by 24 quench protection stations distributed around the ring. They will constantly analyse the voltage transducers connected at each coil access point in order to determine if a quench has taken place. The quench station will then take a series of time critical actions including firing the beam dump and firing the magnet heaters to spread the quench evenly over the magnet. This forces the current to flow through the cold diode protection shunts to avoid local damage to the coil at the quench position. Helium pressure relief valves are then opened to avoid reaching emergency over pressure. A 5 second pre-quench record will be held for "post-mortem" analysis.

The magnet control computer will then take over, switching in series resistors in appropriate places to run the current down as quickly and safely as possible. Constant surveillance of the quench protection stations will of course be necessary as no malfunction can be permitted. Also the state of each magnet will have to be monitored for abnormalities in temperatures and voltage drops to detect deviations which might indicate variations in performance from the model, or incipient trouble. Before each injection extensive automatic checks will have to be done on all magnets, cryogenics and quench protection stations to ensure a fully normal situation.

All of this software will have to be ready and well tested before any significant number of magnets can be put into operation. For this reason the controls effort will liase closely with the magnet test strings and test stands right from the beginning.

IV. BEAM MONITORING

The two main subsystems are the orbit measurement and the beam loss monitors. The monitors will be wired to 24 concentrators round the ring for local processing and connection to the network. Each local unit will have a direct connection to the beam dump which will be activated if conditions which will lead to a quench are detected. In the case of some critical beam loss monitors, like those near the collimators, the dump will have to be activated in 1 or 2 revolutions, ie. 90 to 180 microseconds.

The closed orbit measurement will be systematically acquired by the modelling server to aid in its 2-10 second update of the power supply currents. In addition to helping compensate for the persistent currents, this will provide a continuous on-line correction of the orbit.

The orbit and beam loss monitors will also be acquired on a regular basis by the central alarm server to help in the detection and avoidance of quench provoking situations.

Other instrumentation is foreseen to measure tune, chromaticity, profile, and dynamic aperture and will also be controlled by local sub-systems. These measurements will be used along with the orbit and loss monitors by the programs which set up the machine for injection, acceleration, and collision. A particularly important role for all the instrumentation will be the pilot injection tests first with a bunch of 10^9 protons, then with a 10^{11} bunch to prepare for the full batch.

The beam monitoring stations will be connected by the fast acquisition system to a beam data server and hence to the modelling server. A first trial of the fast acquisition is planned in the near future using some of the LEP beam instrumentation as a prototype.

V. CRYOGENICS

The number of pieces of cryogenic equipment to be controlled is very large. However, the equipment involved is well known to industry, and an industrial control system will be purchased for this purpose. This must communicate effectively with the main control system, as any magnet whose cryogenic state or pre-history is inadequate may be prone to a quench before full field is reached. Thus the state of each magnet must be checked by the control system before each cycle and continuously monitored thereafter. Also access to the cryogenics systems should be the same for normal operator use as the access to any other system.

VI. OTHER SYSTEMS

At present it is assumed that most other LHC subsystems will be controlled by extensions of the LEP control system. If after further study it is found that some other system, for example the collimators, needs special fast control, they can be dealt with using the fast techniques used for the beam monitors and power supplies.

At first LEP and LHC will not function at the same time and therefore much of the standard LEP controls may be re-used for LHC. In any upgrade of LEP this will be borne in mind and spare capacity will be provided where appropriate.

VII. LHC DATABASE

A database (ORACLE) will be used to store the characteristics of the LHC machine and to help with the planning and installation. Such a facility proved invaluable for LEP. For LHC there will be even more high technology equipment to be installed in the tunnel in a shorter time and with limited staff. A maximum amount of easy to use informatics must be provided to ease and control this installation.

All controls information will also be stored in a database. The use of an on-line database was pioneered at the PS on the original IBM 1800, followed by the data-module concept at the SPS. The usefulness of having cabling and installation data on-line was illustrated in the '70's with the SPS experimental area system[3]. For LEP it was necessary to copy the data to local files for use in the control system. We hope to avoid this in LHC and use direct on-line data-base access. Not only will this save work and additional software, but it will reduce the possibilities of errors and inconsistencies. Two approaches are envisaged. For interactive use the database can be interrogated as required by SQL commands during the course of the applications programs. Tests at the PS have shown that response times consistent with human operator interaction can now be achieved. Alternatively, applications programs which must run fast or which need a lot of data, can interrogate the database at start up time and store the data in RAM structures for immediate and fast use, as is done at Isolde[4]. This also protects the programs from inconsistent updates. For any updates to become effective, the program has to be stopped and restarted.

VIII. ARCHITECTURE

Controls technology sees major revolutions on something like a five year time cycle. As LHC completion is more than 5 years away it is premature to make final decisions on the architecture, even more premature for the detailed components to be used. Nevertheless, since this is a controls conference, an architecture is presented for discussion which illustrates much of the current thinking[7]. This is shown in figure 1.

A. Equipment Connection Policy

A major feature is the equipment connection. An important change in equipment connection policy and technology took place in LEP. The responsibility of the equipment layer of the controls was transferred to the groups responsible for the equipment, and a "thin" connection made to the control system. This policy will be continued and reinforced for LHC . The controls group will provide the hardware connection for data and timing, and together with the equipment group it will define a control protocol which will define the characteristics of the equipment to the control system. This policy has several impor-

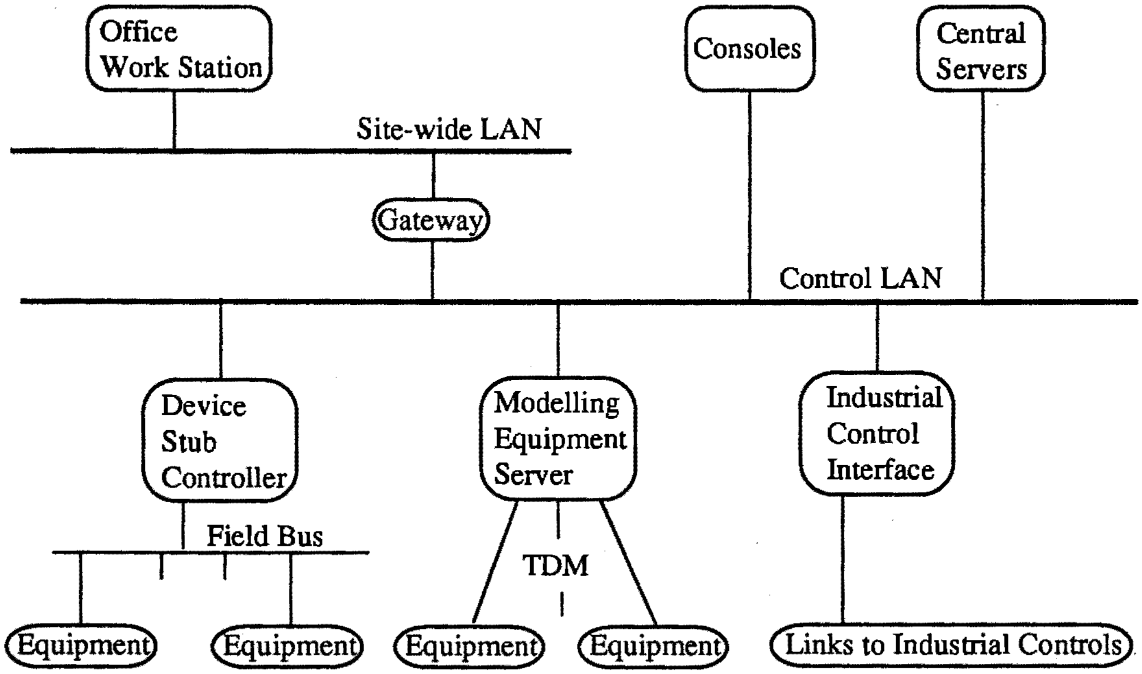


Figure 1. Schematic diagram of architecture

tant advantages. Firstly it clearly defines the budget separation between the control and equipment groups. Secondly it clearly delineates the responsibility for design, performance and maintenance of the control equipment. Thirdly it decouples the evolution of the control system from the evolution of the specific equipment controls. This is becoming an ever more important consideration as the park of control equipment increases and we have not the resources to change everything at the same time.

B. LEP Equipment Connection

The bottom left of Figure 1 shows the LEP equipment connection[5]. The specific equipment is connected to a field bus, MIL 1553, which is connected to a "Device Stub Controller" consisting of an Olivetti PC running the SCO UNIX operating system (replacement by LynxOS, a real time POSIX compliant operating system is planned in the near future. This operating system can also run in a VME crate). These PCs are located in the field, and are linked to the rest of the control system by the control LAN. Much of the LHC equipment will be in common with that of LEP. This includes, of course, all the tunnel management services. Some of the new equipment of LHC may also be connected to the MIL 1553 to save money and development cost. As LEP and LHC will not operate at the same time, economies can be made in sharing equipment and resources.

C. Fast Equipment Connection

The equipment connection shown in the centre bottom of figure 1 is a new development for LHC, being done in conjunction with the SSC[2]. The driving force for this is a fast connection to allow the rapid measurements and power converter updates required for control of the magnetic field. Other advantages of low cost and simplicity are also claimed for this new method, however, which may make it a contender for applications which do not need its speed. The basis of this method is to use Time Division Multiplexing (TDM) techniques to provide an individual channel from each equipment to a central server, all connections working in parallel with no software protocol overhead. This TDM technology is in commercial use for telephone and higher speed data-link requirements by the communications industry, mostly running on optical fibre. It will also be used in LHC for dedicated connections, telephone and video links, interlocks, and many other applications which are traditionally implemented using hard wired copper. Between the equipment and server, the TDM channel will link "reflective memory" cards in which part of the equipment memory containing the control protocol is reflected into the central equipment modelling server. Thus the modelling server can read its input, do its calculations, and write its output with no degradation in performance due to software protocol overheads and long transmission delays. Several servers may be used, eg separate ones for beam monitors, power converters and magnet monitoring. The TDM may be used to link these servers among themselves using reflective memory, or to link the equipment to several servers.

D. Industrial Equipment Connection

The third mode of equipment connection, shown diagrammatically on the bottom right of figure 1, is connection through industrial control equipment[6]. For the cryogenics equipment, and perhaps others, it makes sense to buy the controls from the equipment manufacturer or an industrial process control manufacturer as he has solved the problems, has all the components in house, and can take charge of the installation and maintenance. The controls thus purchased may stop at the level of the Programmable Logic Controller (PLC) for the equipment concerned, or even a complete system may be bought with the manufacturer's network system and consoles so that he can take full responsibility. In both cases an interface to this manufacturer's equipment has to be provided to achieve two way communication and control, and to integrate the industrial controls into the protocols and methods of the rest of the controls.

E. The Central Servers

In addition to the modelling server connected to the power supplies, there will be other central equipment servers connected by TDM to subsystems requiring fast response such as beam monitors, quench protection stations, collimator subsystem controls, dump, etc. Other central utility servers, as shown on the top right of figure 1 will be required for alarms, a file server, an on-line database engine, etc.

F. The Consoles

These will be standard large screen workstations. The PS and SPS console facilities are being upgraded to match LEP[7]. A considerable effort is being put into this and we can assume that the resulting facilities will be entirely adequate for LHC. There is a strong desire to have a single control centre with standardized facilities for the whole of CERN on the LHC time scale.

G. The Office Workstations

LHC will involve a large fraction of CERN's accelerator community over the construction and commissioning period. These people will be spread over a large geographical area. To permit them to work without too much travel to the central control room, access will be provided from the office workstations connected to the site wide LAN. A gateway will be used to authenticate access permissions, and to cut off access at critical times.

H. The Control LAN

A single LAN is shown diagrammatically in figure 1. The present system uses both Ethernet and Token Ring in a bridged configuration. It is reasonable to expect the use of FDDI in the LHC time scale.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S03SRD01

IX. PROTECTION

LHC will have severe requirements concerning the authorization of actions through the computer system. In situations where a quench is possible, equipment changes must be restricted not only to authorized persons but also to authorized programs which have been vetted to perform adequate checks for quench avoidance.

There will be a large number of access points to the system for testing, maintenance and commissioning including, as mentioned above, the office workstations. These numbers will aggravate the protection problem. Network management will have to provide access permission or denial facilities, as now being added to LEP, and the alarm and surveillance system will have to detect and locate abnormal access attempts. The protection will have to be closely linked to the machine operational state. Widespread access will be needed to speed installation, testing and repairs, moving swiftly to a closely controlled situation when quenching becomes possible.

X. TIMING

In addition to data transfer, many types of equipment need special timing signals and events. General purpose timing systems have been developed for PS and SPS from different historical backgrounds. Currently work is progressing on new designs which will cover both areas and which will take into account LHC requirements. Thus the timing distribution for LHC should be part of a new overall system for all CERN accelerators. There is a hope that the timing information may even be combined with the data stream so reducing the number of cables, connectors, and chips, so reducing cost and increasing reliability.

XI. APPLICATIONS PROGRAMS

The maxim "a control system is only as good as its application software" is as true now as ever it was. In the '60s the term "software barrier" was coined to express the difficulties in achieving good applications. In the '70s a determined effort using NODAL in the SPS, PS, PETRA and TRISTAN achieved a good measure of success. In the '80s the problem re-appeared with more complex systems and higher demands. For LHC a serious effort will be made to put together a team of machine physicists and programmers who will ensure that the applications programs will be available, not only to run the machine but also to make a positive contribution to a fast, smooth and safe commissioning.

For LHC, applications software will be required on two levels, at the server level, and at the console level. In the servers sophisticated software will be required for the modelling and real time closed loop control of the high intensity beam in the superconducting magnets. High reliability will be required in this software, and in the quench protection and magnet surveillance programs. All changes and machine settings will have to be verified through a model before being applied so as to avoid quenches.

In the consoles a wide range of software will be required. Automated injection procedures will be needed for speed and reliability. These programs and the server software will have to be ready before commissioning, unlike the cut and try approach which could be used with earlier less sensitive accelerators. Extensive software for beam measurements and diagnostics will be required.

Surveillance and alarm programs will be particularly important to avoid quenches and warn of quench provoking situations. A variant of this software will be required to check out the machine before injection, or before any action which might result in a quench.

Some of this software will have to be professionally written, installed and tested, especially in the servers. Other parts will be better written by the machine physicists and hardware specialists themselves. All will have to be carefully specified beforehand. Tools will include professional development facilities for compiled programs, the old tried and tested NODAL[8], mass market packages such as spreadsheets linked to the machine variables as used at Berkely[9] and in Isolde[4], and other commercial synoptic packages for control as used in LEP.

XII. REFERENCES

[1] Design Study of the Large Hadron Collider (LHC), CERN 91-03.

[2] C.R.C.B. Parker, M.E. Allen, H.C. Lue and C.G. Saltmarsh, A New Data Communications Strategy for Accelerator Control Systems, Nuclear Instruments and Methods in Physics Research, 1990 (Proceedings of Vancouver Controls Conference, 1989).

[3] H.W. Atherton, M. Rabany and R. Saban, A database as a bridge between hardware and software, Proceedings of the 5th International IASTED Symposium, Tunis 1982

[4] R. Billinge, I. Deloose, A. Pace and G. Shering, A PC Based Control system for the CERN ISOLDE Separators, Proceedings of this conference.

[5] P.G. Innocenti, The LEP Control System, Nuclear Instruments and Methods in Physics Research, 1990 (Proceedings ofVancouver Controls Conference, 1989).

[6] M. Rabany, Interfacing Industrial Process Control Systems to LEP/LHC, Proceedings of this conference

[7] R. Rausch and C. Serre, Common Control System for the CERN PS and SPS accelerators, Proceedings of this conference.

[8] M.C. Crowley-Milling and G. Shering, The NODAL system for the SPS, CERN 78-07.

[9] S. Magyary, M. Chin, C. Cork, M. Fahmie, H. Lancaster, P. Molinari, A. Ritchie, A. Robb, C. Timossi and J. Young, The Advanced Light Source, Nuclear Instruments and Methods in Physics Research, A293, 1990.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S03SRD02

A Performance Requirements Analysis of the SSC Control System

S.M. Hunt, K. Low
SSC Laboratory [*]
2550 Beckleymeade Ave.
Dallas, Texas 75237

Abstract
This paper presents the results of analysis of the performance requirements of the Superconducting Super Collider Control System. We quantify the performance requirements of the system in terms of response time, throughput and reliability. We then examine the effect of distance and traffic patterns on control system performance and examine how these factors influence the implementation of the control network architecture and compare the proposed system against those criteria.

I. INTRODUCTION

The Superconducting Super Collider Laboratory (SSCL) is a complex of accelerators being built in the area of Waxahachie, Texas. It will be fully operational at the end of the decade. The SSCL consists of six accelerators: a 1 Gev Linac, three booster synchrotrons (the 12 Gev low energy booster, a 200 Gev medium energy booster, a 2 Tev high energy booster) and two intersecting, contra-rotating 20 Tev synchrotrons that make up the Collider itself. The complex will occupy approximately 112 km of underground tunnels. There are estimated to be about 150,000 control points requiring remote control and interrogation in order to operate the accelerator and diagnose its condition.

II. PERFORMANCE REQUIREMENTS

A. Response time

The control system's response to operator requests should be such that response delays be unnoticeable to the operators. The minimum response time of the control system should be, in the absence of any other constraining factors, 20Hz.
In addition, it will be necessary to provide for higher rates, up to 1 Khz for some essential services like the quench protection monitors (QPM).

B. Throughput

In the Collider tunnel there are 5 Superconducting dipole magnets per half-cell and 968 half-cells per ring. Every 450m is an equipment niche (alcove) which controls 5 half-cells (200 niches). The HEB has 280 half-cells controlled by 24 Niches. The MEB consists of 200 half-cells controlled from 8 surface buildings and the LEB has 108 half-cells controlled from 6 surface buildings. Throughput requirements vary widely [Table 1]. The Linac is not considered here. The environmental (ENV) figures include niche temperature, power, smoke alarms, oxygen and water. The value in the column marked Locations indicates the number of Niches or equipment buildings for a particular machine. The value in the column marked Bytes indicates the number of bytes of raw data being generated at that location for each time interval indicated in the column marked Rate, which is the number of time intervals per second. The value in the column marked Bandwidth is the total number of bytes per second generated for that equipment type and is the product of the Locations, Bytes and Rate values. For the LEB BPM data rates have been set in this table at one tenth of the raw rate in order to conserve bandwidth.

The total amount of data generated site-wide by the SSCL is in excess of 250 Mbytes per second (2 Giga bits per second)[1,2].

C. Reliability

Total allowable unscheduled downtime for the control system is 30 hours in 4505 hours of operation per year. The control system will consist of 205 equipment locations consisting of 162 Collider niches, 24 HEB niches, 11 MEB buildings, 6 LEB buildings, the Linac and the control room complex. Each of these locations will have one communications element (Hub Gateway or multiplexor, depending on the communications architecture chosen) and up to 9 equipment crates.
If each of these 2050 elements (205 locations x 10 elements) is a "critical" system, then to achieve 30 hours of unscheduled downtime with a mean time to repair of 1.5 hours (20 incidents per year) each element would have to achieve a mean time between failure of 54 years [3]. It is therefore clear that other measures, such as the use of redundant systems, will be necessary in order to achieve the necessary reliability figures.

D. Capacity

The installed system should have a capacity at least 50% greater than the requirements stated above. It should furthermore be capable of being expanded by 400% without incurring any additional civil engineering costs or replacement of existing components, only expansion costs.

III. OPERATIONAL REQUIREMENTS

A. Data Accessibility

---

[*] Operated by the Universities Research Association, Inc., for the U.S. Department of Energy under Contract No. DE-AC02-89ER40486.

Table 1. Throughput and response time requirements

System	Locations	Bytes	Rate	Bandwidth
LEB				
BPM	6	108	50,000	32,400,000
RF	1	2,125	20	42,500
RAMPS	6	432	720	1,866,240
VACUUM	6	288	20	34,560
ENV	6	22	20	2640
MEB				
BPM	8	150	6,600	79,200,000
RF	1	2,125	20	42,500
RAMPS	8	600	720	3,456,000
VACUUM	8	400	20	64,000
ENV	8	22	20	3,520
HEB				
BPM	24	70	24,000	40,320,000
RF	1	2,125	20	42,500
QPM	24	60	720	1,036,800
CRYO	24	326	20	156,480
RAMPS	24	280	720	4,838,400
VACUUM	24	187	20	89,760
ENV	24	22	20	10,560
COLLIDER				
BPM	200	58	3000	34,800,000
QPM	200	60	720	8,640,000
CRYO	200	135	20	540,000
RF	1	1062	20	21,240
RAMPS	200	332	720	47,808,000
VACUUM	200	155	20	620,000
ENV	200	22	20	88,000
TOTAL				256,123,700

Control loops will be implemented at the Local (Niche), Regional (Sector) and Global (Control room) levels[4]. However, in order to debug the systems it will be necessary to open some of the local and regional loops from the control room, and move some of the loops from local to regional to control room and vice versa. For instance a control room algorithm might be tested at the global level and then installed as a local loop for reasons of security, because it may need to continue to operate when the global system is in a maintenance mode.

This leads to a requirement that all raw data that might be needed at any level of the control system, even local loops, must be available in the control room. Furthermore this has the important advantage that application programs will have access to all of the data associated with the sensors that might otherwise be hidden. For example if a beam monitor system provided only the result of a calculation, for instance the beam tune, it would not be a simple matter to add another capability, for instance a calculation of the beam lifetime. More importantly new algorithms could not be tested without affecting the operation of existing systems. In addition, as new capabilities are added to systems, the control system should not limit the raw data that is acquired to some arbitrary fraction of the total. All data should be available at all levels.

B. Traffic Patterns

As has been stated earlier, data generated at equipment locations (niches and buildings) should be available simultaneously at more than one location, for instance at the regional level and at the control room to be consumed by console applications. It is not anticipated that there will be significant Niche to Niche communications. This must not however be excluded. Although a number of computers must be able to read the data from equipment locations, only one should be able to write commands to field equipment at any given time. An arbitration mechanism to give permissions to access equipment for commands will be necessary. When command messages are sent to an equipment location it may be necessary to queue the messages for processing. This can be achieved by use of a high level communications protocol such as TCP/IP, but this introduces a large overhead for data transmission of up to 50%. It might be more efficient to queue the requests to the arbitration mechanism that allows commands to be sent to the equipment.

C. Determinism

Application programs typically periodically request current readings and settings from field sensors. It will be necessary to quantify the rate of such requests and guarantee the timely transport of data through the communications systems. The performance of the system must be predictable, that is deterministic. Furthermore because the system will be designed to worst case scenarios, no advantage would be gained by trying to achieve best case performance better than worst case. Thus data transport should be load-independent.

D. System Software

This predictability in the control system should extend to computer operating systems as well as communications equipment. This may mandate the use of real-time operating systems. This would be true for embeded systems using, perhaps, real-time kernels and also for the computers in which run application programs. For these therefore operating systems such as UNIX would be discarded in favour of, for instance, a real-time UNIX.

E. Data Stores

It will be necessary to provide a mechanism to store any data that is acquired to equipment locations or applications programs. This data storage might be for temporary use in shared memory, in a database or a system. As some of this data may be archived for many years it is important that it be stored in a format that does not become obsolete with new versions of the storage utilities. There should be one coherent set of library routines and utilities that can handle these functions. These utilities and library routines must be able to access the structure of the data, not just handle the data set as a whole. The ability to access the data should not depend on the availability of header files that were written at the time with the original application program, and may now be out of date. And they must not need prior knowledge of the structure of the data. The description of the data should therefore be in a standard format, embedded in the data or in a database.

IV. POSSIBLE COMMUNICATIONS ARCHITECTURES

A. SERIAL CAMAC HIGHWAY

Serial Camac meets many of the requirements of a controls communications architecture, all the data is available from a global level, with no data-hiding, but it is difficult to control access to the equipment. Response times and throughput values using present systems are below the level of performance that we require.

B. CSMA/CD

IEEE Standard 802.3 is a "listen while transmitting" LAN access technique, commonly referred to as CSMA CD (carrier sense multiple access collision detection). It closely resembles Ethernet with some minor changes in packet structure together with an expanded set of physical layer options. Bit rates supported are between 1 Mbps and 20 Mbps. Collision detection means that collisions i.e. simultaneous bids for access to the medium can be detected early in the transmission period and aborted, thus saving channel time and improving overall channel utilization. When a collision is detected, the user backs off and continues to attempt access until a maximum number of unsuccessful attempts has been reached before generating an error. A contention-based protocol like Standard 802.3 is unsuitable for a number of reasons. It is not deterministic. It can be totally blocked with no critical data able to get through from a critical system. It is not easy, using standard protocols, to arrange that many stations be able to receive the data, and the data producer has to know the address of the recipient. The access protocol works only for short segments of network, requiring bridges between physical networks. Typically, networks which are lightly loaded with random traffic requests are especially suited for CSMA/CD schemes.

C. Token Rings

The token ring protocol specified by IEEE Standard 802.5 is a polling-based controlled LAN access technique. A station gains the right to transmit on the ring when it detects and subsequently captures the circulating token. This station continues to transmit until it either exhausts all transmission frames or the token times out. The station relinquishes monopoly of the ring by generating a new token which other stations may then acquire. The timing-out mechanism ensures that other stations on the ring have a chance to transmit.

Token passing protocols can be made to be deterministic, but many of the higher level protocols do not take advantage of this feature. Token rings transmission rates can only go as high as 4 or 16 Mbps, meaning that many rings would be needed to achieve the bandwidth requirements of the SSCL. Also, response time is slow due to delays introduced by each station. It is more suited to larger transfers such as file transfers.

D. FDDI

The fiber distributed data interface has the advantages of the token passing protocol and is much faster (100 Mbps). It can operate over the distances covered by the SSCL.

The delay introduced per station is much lower than for token ring as it does not capture all of a packet before retransmitting it, but captures the token while at the same time retransmitting it to the next station on the ring. If the station is the intended destination, determined by examining the first bytes of the token, the outgoing transmission is aborted and this invalid packet is stripped off of the ring by the station that originated it.

However the use of only a single token on the ring at any time means that for large rings such as at the SSCL, when transmitting small amounts of data, most of the time is wasted transmitting the tokens.

E. TDM

Time Division Multiplexing (TDM) is widely used in the Telecommunications industry. It is the method used for passing voice and data channels over common copper and fiber optic media. The system consists of a network in which a channel that is a multiple of 64 Kbps or 1.544 Mbps is assigned between two geographical locations [5]. There may be many channels between these two locations, each channel carrying specific information with all the channels sharing the same fiber media.

The TDM system with its established industry standards of supported transmission rates will be able to address our requirements as outlined below.

V. TDM PERFORMANCE

A. Response Time

The equipment employed has very low overhead, typically 10 μs per node which is less than the speed of light delay imposed by the distance around the Collider. Since it is a point to point network and not a ring, the time to transmit a message is halved as the message does not have to return to its source.

B. Throughput

Throughput is determined by the number of channels assigned to any link. Standards exist for TDM equipment at a number of data rates. Low speed systems use asynchronous transport at (for example) 1.544 Mbps (called T1 or DS1) and 45 Mbps(called DS3).

At higher rates, synchronous systems often based on fiber optic technology are available. These are defined in the Synchronous Optical NETwork (Sonet) standard. This standard specifies rates that are defined at 55 Mbps (called OC1) which can transport 28 DS1 signals and multiples of that rate. The rates are not exact multiples as some overhead information passed for network management which uses the same number of bits regardless of the link speed. OC3 for example can handle 84 T1 signals and runs at 155 Mbps. Commercial equipment available now is capable of transporting 2.5 Gbps (OC48) over a single fiber optic cable.

Work is continuing with multiplexors at OC192 that should be commercially available in a few years, but additional fibers can always be utilized for increased bandwidth. Standard fiber optic cables consist of up to 250 fibers in a 3/4" diameter cable, giving present total capacity of 625 Gbps.

C. Reliability

This type of equipment is used throughout the commercial telephone system where an interruption of service could be disastrous. As this equipment is (as in our case) installed in remote locations, many network management and diagnostic features are built in.

Redundancy is an important feature in these systems. Dual redundant optics and power supplies, and the possibility of building redundant ring networks, are standard features. The use of redundant rings affords network integrity. Data is automatically rerouted the opposite way round the network should the fiber break or otherwise fail.

D. Data Accessibility

As each data channel is independent of all other data channels, there can be no contention in the network. Data from embedded systems or regional computers can be made globally available as the network will have the capacity to handle all data.

VI. TDM IMPLEMENTATION

A. Interface to front end equipment

In each of the equipment locations, for instance Collider niches [Fig. 2], a number of different systems have to be interfaced to the TDM communications system. These include beam monitors, quench protection, ramp generation, vacuum and cryogenics.

These systems will use VXI, VME, STD or CAMAC bus standards or may in some cases be acquisition and control interface cards directly connected to the TDM network. Each of these systems would normally have a 64 Kbps interface to the TDM network. Where a higher interface rate is needed, it would be a multiple of 64 Kbps or a full T1 interface at 1.544 Mbps (24 individual 64 Kbps channels).

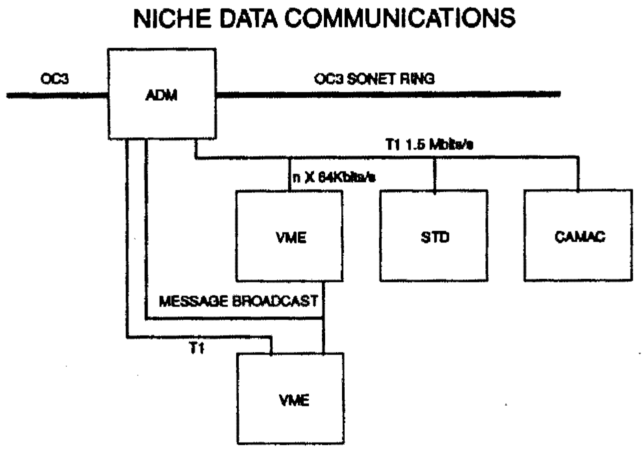


Figure 2: Collider niche data communications

B. Message broadcast system

Some systems also have an interface to a message broadcast system[6]. This uses a T1 interface to distribute, from the control room, medium speed synchronization signals (called events). These signals are characterized by being repetitive such as the 720 Hz clock event or signals that are needed in many locations such as an injection warning event.

C. Long distance links

At each equipment location, the low speed T1 signals are multiplexed using an add-drop multiplexor (ADM) onto a high speed OC3 Sonet link [Fig. 1]. This link will connect all equipment locations in a region. A region would be a Collider or HEB sector (the Collider has 10 sectors, the HEB 2), the Linac, LEB or the MEB. At one location in each region the OC3 link will interface to a global OC48 (2.5 Gbps) Sonet link. This will be a Sonet ring that will connect together all the regions and the control room.

D. Interface to regional computers

The regional computer would also be interfaced to the regional OC3 link. This is to allow it to control regional control loops if necessary. Data arriving from equipment locations would be available to the regional computer as well as transported to the central control room via the OC48 link.

E. Interface to Functional computers

In the control room functional computers running accelerator applications programs will need to have access to data arriving from the equipment locations over the Sonet links. The physical attachment to the TDM networks will be from VME-based Sonet interfaces running

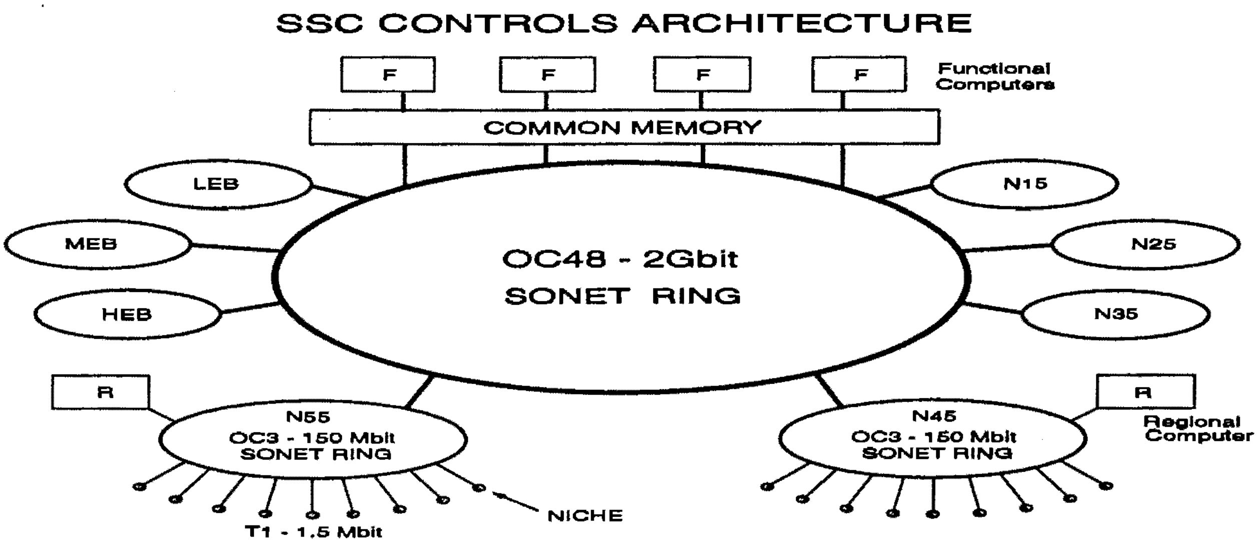


Figure 1: SSC Controls Architecture

at OC3. These OC3 signals will be obtained from an add-drop multiplexor on the OC48 ring. Each functional computer will have access to the data, not only from the OC3 to which it is directly connected but also from all other functional computers. The data arriving will be memory mapped into the virtual address space of all of the functional computers.

VII. CONCLUSIONS

The SSCL performance requirements appear to be attainable with today's technology. Furthermore, the communications network will be largely commercial, thus meeting the reliability and inevitable future capacity upgrades. TDM technology is well-understood and well-established and would not become obsolete during the lifetime of the project.

VIII. ACKNOWLEDGMENTS

The choice of implementation using parallel communication and memory mapping was influenced by the control system of the Advanced Light Source at Lawrence Berkeley Laboratory[7]. Many of the original concepts were proposed by C. Saltmarsh (SSCL) and C.R.C.B. Parker (CERN) who is pursuing some of these ideas for use in the LHC control system at CERN.

References

[1] Site Specific Conceptual Design, SSCL-1056

[2] M.E. Allen, C.G. Saltmarsh, S.G. Peggs, D.A. Dohan, and V.E. Paxson, "Demands Placed on the SSC Laboratory Control System by the Engineering and Operational Requirements of the Accelerators," International Conference on Accelerator and Large Experimental Physics Controls Systems, 1989, Vancouver, Canada.

[3] K. T. Dixon and J. Franciscovich, "SSC Accelerator Availability Allocation", 1991 International Industrial Symposium on the Super Collider, Atlanta, Georgia.

[4] M. Allen, S.M. Hunt, H. Lue and C.G. Saltmarsh, "A High Performance Architecture for Accelerator Controls" , 1991 International Industrial Symposium on the Super Collider, Atlanta, Georgia.

[5] ANSI T1.102,"DS1, DS1C, DS2 and DS3 Levels of Digital Hierachy."

[6] K. Low and R. Skegg, "The Prototype Message Broadcast System for the SSC", IEEE Transactions on Nuclear Science vol. 38, No. 2, pp325-328, April 1991.

[7] S. Magyary et al, "The Advanced Light Source Control System ", Nuclear Instruments and Methods in Physics Research, A293(1990) 36-43, North Holland.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S03SRD03

The Computer Control System for the CESR B Factory

C. R. Strohman, D. H. Rice and S. B. Peck
Laboratory of Nuclear Studies
Cornell University
Ithaca, New York 14853*

Abstract

B factories present unique requirements for controls and instrumentation systems. High reliability is critical to achieving the integrated luminosity goals. The CESR-B upgrade at Cornell University will have a control system based on the architecture of the successful CESR control system, which uses a centralized database/message routing system in a multi-ported memory, and VAXstations for all high-level control functions. The implementation of this architecture will address the deficiencies in the current implementation while providing the required performance and reliability.

I. INTRODUCTION

CESR-B is an upgrade to the existing CESR facility.[1] The major part of the upgrade is the addition of a second storage ring in the existing tunnel. The two rings will operate with asymmetric energies (3.5 GeV and 8 GeV) and will intersect within the CLEO detector. The design luminosity is $3\text{x}10^{33}\text{cm}^{-2}\text{s}^{-1}$ which will be achieved with 230 bunches in each ring.

The control system for CESR-B is also an upgrade of the existing control system.[2] The architecture is shown in figure 1. The MPM (Multi-Port Memory) contains the database and is accessible by the high-level computers and the BCCs (Bus Control Computers). The high-level computers are used to develop and run programs which interface with the operators and physicists to control and monitor the experiment. The BCCs manipulate and move data between the database and the accelerator hardware. Both the MPM and the interface hardware are mapped into the memory space of the BCCs. They only transfer data when requested to by the high-level computers.

It is important to remember the difference between the architecture of a system and how it is implemented. Within a well defined architecture, one can make hardware or software changes to improve some aspect of performance without affecting systems which are outside of the boundary of the control system.

II. DESIGN PROCEDURE

Having decided that the current CESR control system is a suitable model for the B factory, we proceeded to analyze the strengths and weaknesses of our system and the different needs of the new system. Part of this process was defining the scope of the control system.

A. Boundaries of the Control System

For a large design project, well defined boundaries are essential. At the boundaries, the needs of other people must be taken into consideration, and the design process requires communication between the designers and the users of the control system. Within the boundaries, the control system designers can do whatever is needed. We have defined the scope of the control system by defining interfaces for application programmers, instrumentation designers, and operators.

Application programmers must be provided a complete, well documented, set of functions which meet their needs. Programmers are not allowed to bypass these functions by using calls to lower-level routines. CESR uses approximately 35 functions.

Designers of instrumentation hardware are provided with a specification for constructing interfaces to the control system. This includes mechanical, electrical, and protocol details. Recommendations that simplify the control system are included, but not required. This encompasses things like avoiding write-only registers and not having read operations change the state of the system.

The actual implementation of the operator interface is a combination of efforts by both the application programmers and the instrumentation designers. However, it is essential to know the needs of the operators when designing the control system.

B. Special Requirements of the B Factory

We need to know what makes CESR-B different from CESR and how these differences affect the architecture and implementation of the control system.

The first question is how much larger will the new system be? At this early date, we do not have all of the details from the various design groups (eg. vacuum systems, magnet systems), but we do have general numbers. Combining this information with the fact that the amount of equipment in the tunnel will approximately double, we determined that the new control system will have roughly twice as many output control points as the current system.

---

*Work supported by the US National Science Foundation

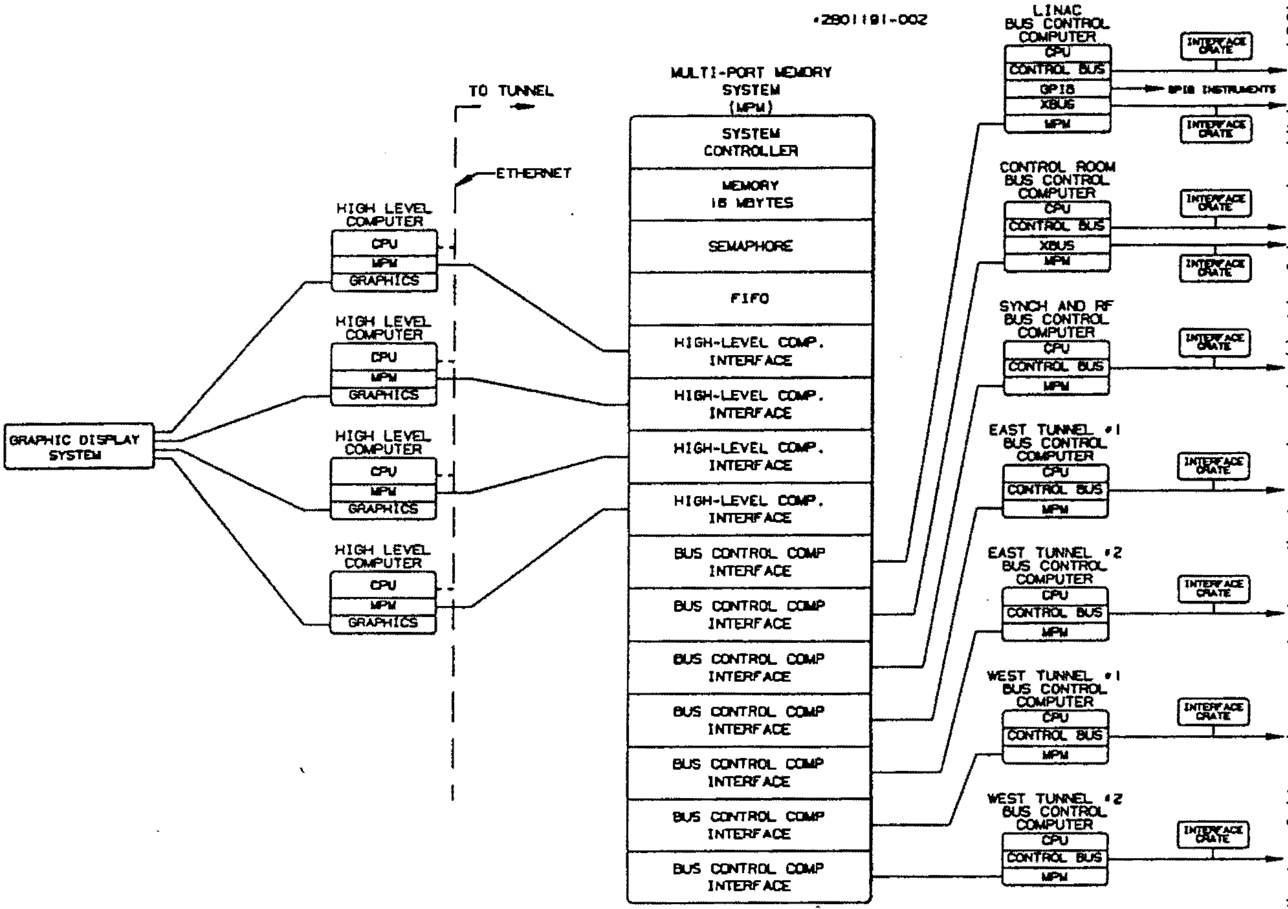


Figure 1. Control System Schematic

We are planning to monitor a great deal more than we currently do. This will help minimize downtime either by indicating when a problem is developing or pointing to the correct area when a malfunction occurs. With the increased monitoring and the additional equipment, we are assuming about five times as many input control points.

There will be several instrumentations systems which require local processors. These systems will need a control system interface for passing processed data. They will also need a connection for downloading and debugging.

C. Strengths of the Existing CESR Control System

Performance is a critical issue. When the operator turns a knob, there should not be a noticeable lag in the response. We run the CONSOLE program at 10 Hz. This is the program that accepts input from the operators and displays results to them. Each time this program runs, it scans the operator input devices, updates the controlled device, and updates the operator's display. It takes about 7 milli-seconds for each pass through this program.

The CESR control system makes very efficient use of the high-level computers. We currently use two VAXstation 3200 computers. The normal application load, consisting of 14 programs, uses 50% of one computer and 30% of the other. This allows special applications (orbit measurements, energy changes) to run in a timely fashion. The efficiency is achieved by minimizing the layers of function calls required to communicate with the hardware and by having processes hibernate when they have requested that the BCCs move a lot of data.

The system is simple and easily extended. We make minimal use of operating system and network functions. This allows us to avoid 'black boxes'; pieces of software over which we have no control. When we added a SUN computer for beam dynamics studies, it was trivial to move the subroutines that are provided to the application programmers.

There is a single database. This allows us to avoid the programs and overhead which would be needed to maintain the consistency of a distributed database.

The database and the interface hardware are memory mapped into the address space of the BCCs. Simple 'move' instructions are used to access the database and the hardware. The control bus can complete a data transfer to the farthest interface crate in less than 15 μsec. Database accesses typically take less than 1 μsec.

Several BCCs can work in parallel on a given transfer request. For example, when reading magnet currents two computers are moving data, one for the east half of the ring

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S03SRD03

and the other for the west half. Three computers control the operator interface.

The graphics display system provides fast and efficient access to many graphics screens, both color and monochrome. It is accessable to all of the high-level computers. Data is sent to it through a FIFO, so the high-level computers simply send data when they have it available.

Geographical addressing is used in the interface crates. Technicians do not need to set address switches on each card. We also insert and remove cards with the power on.

In the tunnel, the entire control system is contained with a metallic exoskeleton for shielding. Feedthroughs are used for connections to the controlled equipment. We are expecting the electrical environment to be more severe in CESR-B.

D. Limitations of the Existing System

The CESR control system has worked extremely well for over a decade. However, it does have limitations, most of which stem from design decisions which were appropriate in the late 1970s.

The address space on the control bus, which connects the interface crates in the tunnel to the bus control computers, is too small. Each control bus has an address space of 65 kwords. This is divided between 16 interface crates, with 16 slots per crate, yielding only 256 addresses per card slot.

We do not have a simple local extension of the control bus. We need to allow designers to build control system interfaces into their equipment and have an easy way to connect to the control bus. Our current system requires two bus operations with a delay of 20 μsec between each operation. A protocol similar to MIL-STD-1553 running in the 5-10 Mhz range would be useful. The length of these extensions would be less than 15 meters.

The interface crates were designed and built by us in the late 1970s. The backplane uses a byte-wide multiplexed protocol on which we transfer one address byte and two data bytes. Since it is a non-standard bus, there is no way to use commercial circuit boards. Producing more crates is very expensive and labor intensive.

There are no facilities for connecting a terminal or a computer in the tunnel. This makes testing and troubleshooting very difficult. We either walk back and forth to the nearest terminal or use the building public address system to communicate with someone at a computer.

The interface from the VAX computers to the MPM is relatively slow. This is due to the fact that the VAX does not have the ability to directly map the entire address space of the MPM. The system uses a set of address and data registers which are located in the Qbus I/O space. To access the MPM, one must first load the desired address into the address register, then transfer the data by reading from or writing to the data register. Since the Qbus has only a 16 bit data path, four Qbus operations are required for each MPM operation. An MPM access requires 12 μsec.

III. HARDWARE

A. High-Level Computers

CESR-B will probably use VAX computers for the major high-level functions. The features of the VAX are that they provide a reasonable development environment, they support priority scheduling of processes, and we are very familiar with them. We have shown that the control system operates comfortably on a VAX, and we expect that the heavier demands of CESR-B can be met by the newer generation VAXes.

As VAX performance improves, the inability to make a memory-mapped interface to the MPM becomes more of a bottle-neck. There is a two-stage plan to address this. First, we will make a 32 bit interface so that setting up the address register and moving the data becomes two operations, instead of four. If even more performance is needed, we will copy the read-only portion of the database into the VAX memory. The VAXes will be clustered to facilitate disk sharing.

There may be other types of high-level computers for special functions. Any computer that we can plug circuit boards into can be interfaced to the control system.

B. Multi-Port Memory System

The multi-port memory (MPM) will most likely be a VMEbus based system, although the Futurebus and any other contenders will be investigated. The MPM contains the RAM used for the database and for message passing. It also contains special hardware which is used to enhance the multi-processor aspects of the system. It can support up to 16 interfaces to high-level or bus control computers.

The system controller in the MPM is supposed to guarantee that under normal conditions no processor has to wait more than 4 μsec for a transfer to complete. This is to satisfy the bus-timeout requirements of the high-level computers, but it means that if the full complement of 16 processors are connected to the MPM, each bus operation must finish in 250 nsec. Several things are done to achieve this. No processor can own the bus for more than one operation. There is neither read-modify-write nor block-mode capability. The system controller contains a special 16-way round-robin arbiter. This uses wiring added to the backplane which provides individual bus request and bus grant lines for each processor interface. The timeout circuit is adjusted according to the needs of the slowest slave device, which is the memory, and must account for the actual access time plus any dead-time from error correction or memory refresh. In CESR, it is set for a 2 μsec period.

CESR uses about 3 Mbytes of a 4 Mbyte RAM board. It is error-correcting memory with a cycle time of 400 nsec. We discovered that the cycle time is more important than the access time in a multi-processor system. One memory board that was supposed to be fast malfunctioned if consecutive accesses occurred too close together. We expect to use between 16 to 32 Mbytes of RAM for CESR-B. The performance of most modern RAM boards will be adequate.

Semaphores are provided for controlling access to shared data areas. The semaphore is a hardware test-and-set register. If the semaphore is not owned, a read operation returns a zero and sets the semaphore to one. If the semaphore is owned, a read operation returns a one. Writing resets the semaphore to zero. There is one semaphore for each longword (4 bytes) of RAM. This simplifies assigning semaphores. One just adds a fixed offset to a RAM address to access the semaphore associated with that address. The semaphores are implemented with a PAL and static RAMs. A 1 Mbit SRAM can make enough semaphores to cover 4 Mbytes of RAM, so it is easy to provide a sufficient number. Semaphore cycle time is 100 nsec.

The FIFO board is provided for passing messages between processors. A single operation can send a message to any number of processors. The FIFO also provides a queue to manage multiple messages to the same processor. The message passing protocol is defined so that the FIFO cannot overflow. Special wiring is provided so that an interrupt can be generated when a processor's FIFO contains a message. A processor can also poll a status register to determine if there is a message. FIFO cycle time is 150 nsec.

C. Bus Control Computers

The BCCs transfer data between the MPM and the accelerator hardware over the control bus, performing a variety of operations on the way. They are supposed to complete each transfer request as quickly as possible, then wait for another request. The computers contain interfaces to the MPM, the control bus, and, in some cases, the CESR Xbus. CESR uses 68020 CPUs and CESR-B will probably use the same family. Each computer runs a single common program; there is not even an operating system on them. The program is written on the VAX in 'C', compiled by a cross-compiler, and downloaded into the MPM. Bootstrap code in the BCCs moves the code from the MPM to local RAM and starts execution.

Under normal load, the CESR bus control computers are idle 85% of the time, which means requests are usually handled immediately. CESR-B will add two more computers to handle the additional equipment, for a total of seven. Boards with 68040 processors should be able to provide the CESR-B control system with enhanced performance appropriate to the heavier load.

The MPM interface passes memory references that fall within a particular address range on to the MPM. The BCCs are in the same physical location as the MPM. In CESR, we use a 32 bit, multiplexed, single ended connection between the BCCs and the MPM. It has an overhead of 300 nsec. For CESR-B, we will provide the same functionality, probably with the same type of interface.

The Xbus interface drives the control bus used in CESR. There are some places where CESR-B might use the same hardware as CESR, for instance in the LINAC or the Control Room, so an Xbus interface is required.

The control bus interface communicates with the interface crates around the lab. The details of this interface will depend on the design of the control bus itself.

D. Interface Crates

Interface crates will be distributed throughout the tunnel and in the control room. In the tunnel, there will be 4 control busses, each with 16 crates. The crates will be configured so as to maximize the effect of the parallel operation of the BCCs. This will involve two busses in each half of the tunnel, with crates alternating between the two busses.

The interface crates will use commercial VMEbus backplanes running the standard VMEbus protocol. At a minimum, they will support short (16 bit) and standard (24 bit) addressing with 1 Mbyte of address space per crate. Data width will be 16 bits. We may choose to support long (32 bit) addressing and 32 bit data.

The interface crates contain I/O devices which we would like to map into the address space of the bus control computers. The crate controller, which interfaces to the control bus, should be a simple design. We do not plan to have a general purpose processor board in each crate.

By using a commercial bus we will be able to buy circuit boards for many functions. However, we will most likely design and build our own interfaces for the more common functions. This will allow us to get exactly what we need, without too few or too many features. Maintenance will be simplified since we will use a consistent design for the bus interface logic.

We want to support geographical addressing, but this requires an extension to the VMEbus specification. Our plan is to divide the short address space between 16 backplane slots, yielding 4 kBytes of address space per slot. We will use four pins on the VMEbus P2 connector. On the boards that we design, these pins will be used to match address bits [A15..A12]. Commercial boards and designs that require more than 4 Kbytes of address space will use the standard (24 bit) addressing mode, with switches or jumpers on the board.

We will also investigate the issues involved with live insertion.

E. Control Bus

The control bus is a data highway between the interface crates and the bus control computers. We have not decided how to implement this connection. The speed and performance should be as good as or better than the Xbus used in CESR. This bus has about 4 µsec of protocol overhead plus a round-trip propagation delay of about 3.5 nsec per foot. It uses differential data transmission for noise control and parity for error detection. At a minimum, we need to transfer 24 bits of address, 16 bits of data, plus some control signals. The three options under consideration are a fully parallel system, an address/data multiplexed system, and a serial system.

The parallel system is logically the simplest. We would just need to buffer the address and data lines of the BCC.

Protocol time could be less than a μsecond and throughput would be limited by propagation delays. The drawbacks are the amount of cabling and the number of connectors and bus receivers. Forty wire pairs are required just for the address and data signals.

The multiplexed system, where the address and data signals share the cable, is almost as simple as the parallel one. The receiving boards would need an address latch, but fewer receivers. Another μsecond would be added to the protocol time for the address transmission phase. It still needs in excess of 30 wire pairs, so the cabling is not trivial, but it is a one-time process. There are commercial products that could satisfy our needs.

A serial system requires only a single conductor. It could be a fibre which would provide noise immunity. Cabling would be simple. The disadvantages are speed and electronics complexity. Moving more than forty bits of data in 4 μsec indicates that the data rate would need to be in excess of 10 Mhz. Serial-to-parallel conversion would be needed at both ends. We are looking at some FDDI chipsets which would simplify this type design.

F. Control Bus Extension

The control bus extension allows designers to build a control system interface into their systems and eliminates the need to bring many analog and digital signals to the interface crate. The extension will provide up to a 4 kbyte address space, which may be shared between several remote devices.

The choices for a bus extension are the same as for the main control bus. Minimizing congestion in the tunnel by minimizing the number and size of the cables and connectors makes a serial protocol highly desirable. We are looking at implementations using MIL-STD-1553, 16 Mbit per second token ring, high-speed UARTS, and TAXI chips.

G. Graphic Display System

The graphics display system will provide each high-level computer with direct access to video displays. There will be at least 4 color display channels and 16 monochrome channels. The channels will be distributed throughout the lab through our video distribution system.

There is a large FIFO connected to each computer. When a program needs to update a display, it allocates the FIFO and sends its data. Aside from the appearance of the graphics, there is no acknowledgement that the data has been transferred. The high-level computer doesn't have to wait.

We will be investigating X-terminals. In particular, we will look at their response time (can we turn a knob and have a reading on the screen track the changes?) and the amount of computer resources that they use.

H. Special Function Hardware

Some applications require a dedicated processor. These systems either handle large amounts of data or need higher update rates (>10Hz) than can be handled by the bus control computers. These include beam position monitoring, beam lifetime monitoring, the collision assurance system, and feedback control equipment. These systems interface to the control system via a shared memory and only transfer data that is needed by the high-level computers. An ethernet will be provided in the tunnel for maintenance functions, such as downloading and debugging.

The GPIB is supported by an interface in one of the BCCs. We do not have any CAMAC plans, but if needed, it too can be driven by a bus control computer.

IV. DATABASE

The database contains all of the information required to define the accelerator hardware and to communicate with it.

The heart of the database is the Name Table. It contains an entry for each node in the control system, where a node is a grouping of related hardware or software entities. The name table entry for each node contains a 12 character mnemonic name, information about how many elements and properties the node has, and pointers into the data area for each property. Properties include control bus addresses, scale factors, and raw and processed data. There is also information about which BCCs are used for a given node and the type of processing that is performed on the data.

Additional data structures are the hash table, the link table, the request packet area, the request packet address table, and the data area. These structures will be described by way of looking at a typical operation.

V. TYPICAL OPERATION

An example of a common operation is reading the quadrupole magnet currents. The application program requests data by making the subroutine call:

```
call vxgetn('CSR QUAD CUR',num1,num2,readout_vec)
```

where 'CSR QUAD CUR' is the mnemonic name for the CESR quadrupole magnet current node, 'num1' and 'num2' are the first and last elements of this node that the user wants to read out, and 'readout_vec' is an array where the data will be returned. This subroutine, like most of the control system routines, will not make any subroutine calls. It will directly communicate with the MPM. We will go through the sequence of operations performed by this subroutine and show how the hardware, software, and database function together. Performance measurements based on CESR will be provided.

A. Initialization

The VAX computers cannot directly map the MPM, but instead use interface registers. The VAX to MPM interface provides 32 sets of registers. When the VAX is booted, the VMS program 'SYSGEN' is used to create 32 dummy devices, one for each register set. With this technique, the operating system handles allocation and cleanup. A program allocates a set of registers and owns them for the duration of its execution.

The SUN computer can directly address the entire MPM address space, so its programs simply use pointers. The mapping registers must be initialized at boot time.

The program also needs to allocate a Request Packet. It does this by reading the semaphores associated with the Request Packet Address Table. When an unowned semaphore is encountered, the program records the number and address of the Request Packet.

B. Search the Name Table

The mnemonic name is hashed by exclusive ORing the three longwords of the name and dividing the results by the size of the Hash Table. This produces an index into the Hash Table. The Hash Table entry, which is a pointer into the Name Table, is retrieved. The requested name is compared with the name found in the name table. If they match, then the search has been successful. If they do not match, then the Link Table, at the same offset, has the index of a new Hash Table entry. The name comparison is repeated.

In CESR, with over 900 nodes in the Name Table, the names will match on the first try 90% of the time. No lookups require more than three tries. A name table lookup requires 100 μsec.

C. Set Up and Deliver Request

The Request Packet owned by this process is filled in. The program inserts the Name Table pointer, the number of the first and last elements desired, and the type of operation that the BCC should perform. It also inserts 'BCCs used' bit field, which identifies which BCCs may be involved. This piece of data is inserted in four entries of the Request Packet; the 'used', 'start', 'done', and 'error' entries.

The 'BCCs used' bit field is combined (ORed) with the number of the Request Packet. The result is written to the FIFO board, which signals the appropriate BCCs that there is work for them. The logic on the FIFO board causes the number of the request packet to be written into all of the FIFOs which have a bit set in the bit field.

At this point, the program waits for the bus control computers to move the data. Programs can either go into a wait loop or they can hibernate. The choice depends upon the programmer and how many elements are involved. Most programs hibernate, which contributes to the efficiency of the high-level computers.

D. Bus Control Computer Operation

The BCCs are normally executing in a loop, checking their FIFO to see if there is a message. When there is one, the BCC reads the Request Packet number from the FIFO. In the Request Packet, the computer clears the bit which identifies it in the 'start' entry. Since there may be several bits set, a semaphore must be used to guarantee that only one computer at a time is changing a bit.

From the Request Packet, the computer gets the name table pointer, the element numbers, and the operation mode. It verifies that all database pointers required for the operation are valid. It uses an entry in the database which specifies the first and last elements that this computer handles to modify the element numbers from the request Packet. This keeps the computer from spending time trying to transfer data related to elements controlled by another computer.

Once all of the checking is complete, data movement begins. The control bus address of the next element is retrieved from the database. If the address indicates that the element is controlled by this computer, the raw value of current is read over the control bus from the magnet and stored in the database. If the element is controlled by another BCC, then this computer goes on to the next element.

Using parameters from the database, the raw value is then scaled and offset. The final result is written to the database. The time and status of the operation are also stored. Error information, if any, is saved. The process continues until the last element is done. Notice that other BCCs may be working on this request at the same time.

After all of the elements have been processed, the computer clears its bit in the 'done' entry of the Request Packet and in the 'error' entry (if there were no errors). Again, semaphores are used when changing bits.

E. Retrieve Status and Data

The high-level computer reads the 'done' entry in the request packet. When all of the bits are zero, then all of the bus control computers have finished. If the 'error' entry is all zeroes, then there were no errors. The data is read from the database and moved into the user's array.

In CESR, elements 1 through 49 of the 'CSR QUAD CUR' node are on one control bus and elements 50 through 98 are on another. The execution time of the 'vxgetn' subroutine to read the current from one magnet takes 700 μsec. Reading 49 currents requires 4780 μsec, or 80 μsec per element. The bus control computers use 50 μsec and the high-level computer uses 30 μsec. When reading all 98 elements, the advantage of the parallel operation of the BCCs becomes obvious. It takes 6200 μsec. The extra 1420 μsec is just the time that the high-level computer needs to move the data for the additional elements into the user's array.

VI. REFERENCES

[1] CESR/CLEO Staff,"Conceptual Design for a B Factory Based on CESR," CLNS 91-1050, 1991.

[2] C. R. Strohman and S. B. Peck, "Architecture and Performance of the New CESR Control System," in the *Proc. of the 1989 IEEE Particle Accelerator Conference*, Chicago, IL, March 1989. pp. 1687-1689.

Standards and the Design of the Advanced Photon Source Control System*

William P. McDowell, Martin J. Knott, Frank R. Lenkszus, Martin R. Kraimer, Robert T. Daly, Ned D. Arnold, Mark D. Anderson, Janet B. Anderson, Robert C. Zieman, Ben–Chin K. Cha, Fred C. Vong, Gregory J. Nawrocki, Gary R. Gunderson, Nicholas T. Karonis, and John R. Winans
Argonne National Laboratory
Advanced Photon Source
9700 South Cass Avenue
Argonne, Illinois 60439

I. INTRODUCTION

The Advanced Photon Source (APS), now under construction at Argonne National Laboratory (ANL), is a 7 GeV positron storage ring dedicated to research facilities using synchrotron radiation. This ring, along with its injection accelerators is to be controlled and monitored with a single, flexible, and expandable control system. In the conceptual stage the control system design group faced the challenges that face all control system designers: (1) to force the machine designers to quantify and codify the system requirements, (2) to protect the investment in hardware and software from rapid obsolescence, and (3) to find methods of quickly incorporating new generations of equipment and replace obsolete equipment without disrupting the existing system. To solve these and related problems, the APS control system group made an early resolution to use standards in the design of the system. This paper will cover the present status of the APS control system as well as discuss the design decisions which led us to use industrial standards and collaborations with other laboratories whenever possible to develop a control system. It will explain the APS control system and illustrate how the use of standards has allowed APS to design a control system whose implementation addresses these issues. The system will use high performance graphic workstations using an X–Windows Graphical User Interface (GUI) at the operator interface level. It connects to VME–based microprocessors at the field level using TCP/IP protocols over high performance networks. This strategy assures the flexibility and expansibility of the control system. A defined interface between the system components will allow the system to evolve with the direct addition of future, improved equipment and new capabilities. Several equipment test stands employing this control system have been built at ANL to test accelerator subsystems and software for the control and monitoring functions.

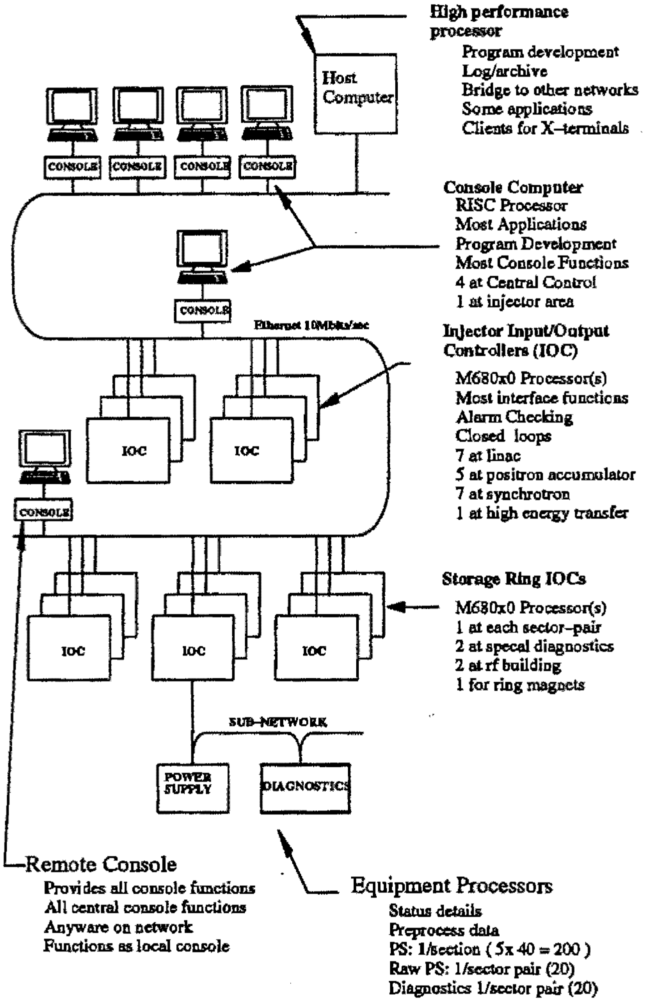


Figure 1. APS Control System

II. STANDARDS AND THE APS CONTROL SYSTEM

The APS control system must be capable of (1) operating the APS storage ring alone and in conjunction with its injector linacs, positron accumulator, and injector synchrotron for filling, and (2) operating both storage ring and injection facilities as machines with separate missions. The control system design is based on the precepts of high–performance workstations as the operator consoles, distributed microprocessors to control equipment interfacing and preprocess data, and an interconnecting network. In a paper presented at the 1985 Particle Accelerator Conference [1] we outlined our initial approach to the APS control system. In this paper we predicted

---

*Work supported by U.S. Department of Energy, Office of Basic Energy Sciences under Contract No. W–31–109–ENG– 38.

that the control system would use workstations for the operator interface, single–board computers to control the front–end electronics, and a network consisting of either Ethernet or Token–Ring. The APS control system today is remarkably close to the initial design concepts due to rapid performance gains in computing workstations, low cost network connections, both Ethernet and Fiber Distributed Data Interface (FDDI), and availability of real–time operating systems for the front–end computers.

Figure 1 shows in schematic form all major components and their relationships. The current design includes about 45 distributed microprocessors and five console systems, which may consist of one or more workstations. An additional 70 Input–Output Controllers (IOC's) will be used to control the insertion devices, front ends, and beam lines.

The operator interface (OPI) is implemented with a high performance graphic workstation and uses an X–Windows based GUI. Standards play a large role in the selection of the OPI since the hardware, operating system GUI, and network must be compatible. The ideal control system design would be vendor independent in all of these areas. To make the APS control system vendor independent we chose to use standards when selecting these components.

A. Standards

The definition of the word "standard" used for the purposes of this paper is as follows: "Something established by authority, custom or general consent as a model or example" [2]. Past practice at large laboratories has been to invent almost everything that was needed to build a control system. Accelerator control system groups have built computer systems and designed networking schemes. Of course there were good reasons for this – the laboratories were often pushing the leading edge of electronic and computer technology and the required devices and techniques were not available on the open market. This picture has greatly changed. Computer technology has now spread into every corner of our lives. There are literally tens of thousands of companies inventing new uses for computers and pushing the limits of technology. This has had a very positive effect on control system design as the effort required to build a control system can now be redirected towards control and accelerator details rather than details associated with building a computer or computer network. In the Proceedings of the Second International Workshop on Accelerator Control Systems [3] held in October of 1985, no discernible trend can be observed in control system design. This contrasts with the sense one receives from reading the titles of the the papers presented at the 1991 Particle Accelerator Conference. These titles show a ground swell towards what could be called a generic control system. The generic system consists of workstations running UNIX, a network, and front end processors running a real time operating system. We now find standards being followed at all levels of control system design.

B. Operating Systems/Workstations

At the Europhysics Conference on Control Systems for Experimental Physics [4] in October of 1987, discussion panels ran late into the night with the "religious" arguments for the choice of UNIX or VMS as the operating system of choice for control systems. There where convincing arguments presented by advocates of both sides of the discussion. Four years later the argument has been settled, not because either of the opposing sides was won over by a technical argument but because of market forces. The development of the the Reduced Instruction Set Computer (RISC) processor has resulted in UNIX dominating the workstation market.

RISC is a recent innovation in computer architecture (although some people claim that the PDP–8 was a RISC machine). The study of computer instructions and their relative frequency of usage revealed that most of a computer's time is spent in the execution of a small subset of its repertoire. RISC architecture takes advantage of this fact by streamlining the execution of this subset and by implementing the less used and more complex instructions with combinations of the (now fast) small set of instructions. Since there is now a small set of simple instructions, parallel "pipelining" can be used to increase execution speed. In this method more than one instruction can be executed simultaneously by staggering in time the various suboperations. Some processors can even average more than one instruction per clock cycle.

The converse to RISC architecture is Complex Instruction Set Computer (CISC) architecture. Most computer architectures developed prior to 1980 are of the the CISC type, a typical example being the VAX. Today there is a five–to–one advantage in raw MIPS (millions of instructions per second) for RISC devices. This should be discounted to some degree since RISC requires more instructions to perform some types of operations, but an advantage of even two–to–one on reasonable benchmarks is obtainable.

The UNIX operating system itself was originally developed by Bell Telephone Laboratories as a word processing tool, but it was soon modified to support software development tools and finally grew into a full–featured operating system. UNIX was written in the "C" language, also developed at Bell Telephone Laboratories. The keys to UNIX's success are that it is extensible and it is written in a portable language. These attributes allow the user to make enhancements, remove features, and tailor UNIX to specific needs. One indication of this is the fact that the UNIX operating system is available for microprocessors as well as supercomputers. Thus, if a start–up company chose UNIX as its operating system and made the changes necessary to support its chosen computer architecture, any existing software that ran under UNIX could be recompiled to run on their computer. In this way new computer architectures can be introduced with ready–made operating system software and trained users.

Because of the development of RISC processors and the existence of UNIX, nearly all computer manufacturers are developing and marketing RISC–based computers and workstations which use the UNIX operating system. Competition is driving performance up while keeping prices low and this trend is likely to continue. There is still a market for CISC architecture computers and operating systems such as VAX/VMS, principally due to the installed base of application software and the steady improvements made to the hardware by vendors.

These reasons seem to make it obvious that the operating system of choice for any control system to be delivered in the mid 1990s will be UNIX. The bottom line for APS is the fact that the UNIX operating system provides the control system a large measure of vendor independence. We have the OPI software running on SUN 4 and Digital Equipment DS5000 workstations and expect to port the system to other vendors' workstations.

The GUI "wars" now being fought in the press and on workstations provide a very good reason to conform to standards when writing the OPI software. APS is developing applications using the Open Software Foundation's Motif toolkit. We are extensively testing the software against the two major window managers Motif and Openlook.

C. Front End Systems

The IOC, or front–end electronics, is implemented with single–board computers of the Motorola 680X0 family, packaged along with signal interface cards in VME and VXI form factor crates. Motorola 68020 processors are used in initial configurations with 68040 processors planned for most future configurations. A real–time operating system, VxWorks from Wind River Systems Inc., is used to provide multitasking, high performance front–end software. More than fifteen VME input–output modules are currently supported. These modules include binary input and output, analog input and output, motor drivers, counter timers, and subnet controllers. More modules will be supported as they become available and are required. Most information preprocessing is performed at this level with only engineering units sent to the OPI for display. Signal monitoring can be set up to communicate only on signal change or limit–breaching or at some preselected rate. Local sequential and control–loop operations can also be performed. In this way, maximum benefit is gained from the many IOC processors operating in parallel. This is one area where APS is vulnerable to complications which would arise if the vendor of the real–time software failed. When the posix standard for real–time systems becomes a reality and most real–time vendors conform to the standard, our estimate is that it would take about two months of a very knowledgeable programmer's effort to change real–time kernels.

In addition to local IOC I/O, subnets are utilized to interface remote points where an IOC may not be present to a distant IOC. There are currently three supported subnets in the APS control system: Allen Bradley, GPIB, and BITBUS. The Allen Bradley I/O subnet uses the 1771 series I/O modules to provide basic binary and analog I/O support for the APS control system. Allen Bradley is a inexpensive and rugged standard for industrial control systems. Copper and fiber optic based multidrop subnets are available for this equipment.

Laboratory test and measurement equipment often use the GPIB standard as an interface to an external control system. This multidrop standard presents some serious challenges and potential problems to the system designer. GPIB has a distance limitation of 20m which requires the instruments connected to the bus to be in close proximity to the IOC. In addition, the signals within the GPIB cable are not balanced and thus susceptible to EMI/RFI noise and ground spikes. Signal transfer and isolation techniques are not part of the GPIB standard and although commercial equipment is available to extend the distance of a GPIB interface, it is prohibitively expensive.

BITBUS provides a method of high speed transfer of short control messages over a multidrop network. The BITBUS subnet can be used as a method for remote, single point I/O as well as a gateway for remote GPIB and RS232 signals [5]. A differential, opto–isolated, wired subnet is the BITBUS standard; however, a multidrop fiber optic network has been developed for BITBUS at the APS.

D. Networks and Protocols

Argonne uses Ethernet as the intra–laboratory network. There are backbone cables in the individual buildings with communication between buildings presently done via the Lanmark PBX system. Intra–building FDDI will be available within 6 months. The control system development computers are presently sharing the APS Ethernet backbone with all other APS computing needs (55 Sun Workstations, a VAX Cluster with six members, 18 terminal servers, 40 PC's using Pathworks, and 40 Macintosh systems). Two test stands and six development IOC's are running in this environment without experiencing network–induced problems. A Network General Sniffer is on line at all times should the need arise to diagnose an apparent problem. In the APS facility, however, we plan to use FDDI with Ethernet branches as performance needs dictate.

The central feature of both the OPI and IOC software designs is the protocol for connections between software modules for the purpose of exchanging information. This protocol is called channel access [6] and is built on the TCP/IP Standard. TCP/IP is an integral part of every UNIX–based workstation as well as being built in to VxWorks, the real–time operating system. When an OPI application program needs to connect to a process variable located in an IOC, it issues a broadcast over the network and the IOC in whose database the requested process variable resides provides a response. A socket–to–socket connection is established and thereafter efficient two–way communication takes place. IOC–to–IOC channel access can take place to exchange inter–IOC information. It should be noted that the OPI needs no knowledge of the location of the desired process variable, only its name. Figure 2 shows the relationship of the channel access software in both the OPI and IOC systems. It also illustrates how a mouse and screen "slider" are used to communicate, through channel access software at both ends, with a D–A convertor. Similarly, A–D convertor output finds its way to a screen.

The database which defines all IOC channel connections and properties is distributed over the many IOC's and downloaded at IOC boot time along with the operating system and the particular device driver software modules required by each IOC. The entire database is centrally maintained and configured with a UNIX workstation which, of course, can be any OPI. Figure 2 shows the downloaded location of the IOC database in the overall data flow.

E. X–Windows

In the X–Windows client–server paradigm, an application program is divided into the "client" (which provides the computation and logic of the program) and the "server" (which provides the interaction facilities for the human operator or user). In the APS control system, both client and server are implemented in processors at the OPI level. The client and server need not reside in the same processor so that, for example, a specialized parallel processor may provide client services for a more common workstation server or X–Window terminal. In this way, the OPI's will be able to have windows open to clients operating both locally and on other processors on the network.

F. Application Software

Application software comprise those programs which the operator or physicist invokes to provide a feature or service not provided by the equipment operation level of the control system. An example would be the software required to provide a local bump in the orbit. These programs can be of two general forms. The first is a control panel which is created during a session with a display editor (see Figure 2, upper left). Graphic tools such as buttons, sliders, indicator lights and meters, and graph paper are selected and located on the panel. Static entities which can be used to depict the physical system, such as piping diagrams, are added where appropriate. Connections to IOC channels are specified at this time and the proper drawing list and action code are automatically generated. When complete, the panel is called up for execution, the channel access calls are made, and the control panel is now "live." No actual code is written or compilation made, aside from that originally involved in the tools themselves. The software provides calculation records and allows cascading of physical inputs and outputs with these calculations. This allows very complex operations to be designed. The second form of application program is that of employing classic in–line code generation. In this case standard entry points are provided to the same graphic and channel access tools. Using this approach, an existing code can be adapted to our system by calling the channel access code and displaying the results using either traditional line–by–line or graphical output.

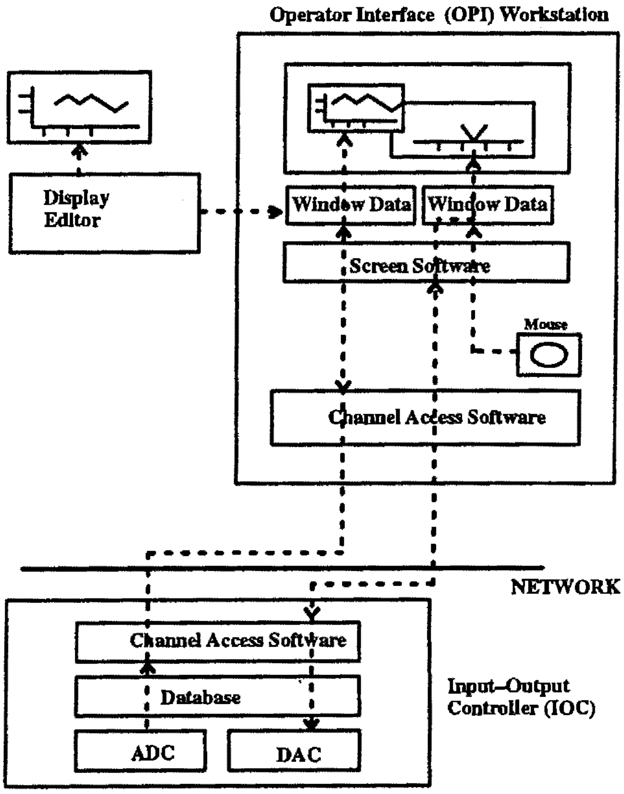


Figure 2. APS Control Environment

All software is being developed under UNIX, including that for the IOC's. In this way, windows can be opened simultaneously at an OPI for software development, actual run–time applications, database configuration, electronic mail, etc. This streamlines software development, database servicing, and system troubleshooting.

III. INTERLABORATORY COOPERATION

At the Accelerator Control Toolkit Workshop [7] in 1988 a group of people responsible for accelerator control systems at laboratories throughout the world spent a week discussing various aspects of control systems. One of the topics discussed was the development of "tools" which could be used at more than one laboratory. Subsequent to this meeting we decided that the APS would pursue the idea of looking at existing control systems with the aim of determining if they could be used at APS. After much discussion we decided to pursue collaboration with Los Alamos National Laboratory (LANL). Discussions were held with the developers from LANL and it was decided that APS would send a representative to LANL who would use the system to develop an application which would be useful to LANL. One of us (Kraimer) spent a summer at LANL developing the software to control a magnet measuring facility. Upon his return he imparted his positive impressions of the system. We then decided to complete an in–depth study of the software. He sequestered himself in his office with the software listings. As he gained understanding of the system he gave tutorials to the APS controls group staff on the internal design of the software. Further group discussions led to the decision to try to form a cooperative development team with LANL. M. Knott (ANL) and M. Thout (LANL) proceeded to develop an agreement that has led to the co–development of the Experimental Physics and Industrial Control System (EPICS). The paper entitled " EPICS Architecture" [8] by L. R. Dalesio, et al. presented at this conference provides a detailed look at the features of EPICS.

IV. CONSTRUCTION STATUS OF APS AND THE APS CONTROL SYSTEM

Construction is proceeding rapidly on the physical structure of the APS. As of this date (late October 1991) the linac and injector buildings have steel erected, the concrete for the linac tunnel and positron accumulator ring vault is in place, the control center has reached the first floor level, and the foundations are in for the first section of the storage ring building which will be used as an early assembly area and magnet measuring facility. Barring unforeseen construction delays, the linac and control center are scheduled for occupancy in April of 1992.

The APS control system is now actively supporting two test stands, rf and linac. Work on these test stands started in 1989. In their first implementation they used a predecessor version of the OPI running on a VAXStation under the VMS operating system and a predecessor of EPICS called GTACS (Ground Test Accelerator Control System) for the IOC. As work on a UNIX–based OPI and EPICS progressed, both test stands converted to the UNIX OPI software and EPICS. The APS rf test stand was reported at the Real Time '91 Conference [9]. Two IOC's are being used to implement the linac functions: one for beam diagnostics and the other for control. The test stands have proved to be highly beneficial to both the controls group staff and the linac and rf systems development team members. The controls group has gained experience in using the control system as well as received suggestions for changes and improvements. The test stand staff has been able to concentrate on linac and rf design details without developing their own control system. The only way provided to remotely run the test stands is via the control system.

Progress in the development of EPICS software is continuing. An alarm handler [10] has been developed and is being optimized. We are continuing to add device and driver support for new hardware modules as well as develop new record types such as pulseTrain, pulseDelay, etc. A graphical database link display tool is being developed as a way to document databases and requirements are being developed for a system–wide database and a system–wide error handler to accept and process IOC–generated errors. We are developing low cost IOC's based on single–height VME modules as well as Gespac G64 modules. VXI crates using the standard VME processors and network boards are currently operating. On the OPI side we are running on both the SUN 4 and DEC 5000 platforms and we will soon port the system to Hewlett Packard 700 series workstations.

V. REFERENCES

[1] M. Knott, G. R. Gunderson, F. R. Lenskzus, W. P. McDowell, "Control System Features of the Argonne 6GeV Synchrotron Light Source," IEEE Transactions on Nuclear Science, Vol. NS–32, No. 5, October, 1985.

[2] Webster's New Collegiate Dictionary, Merriam, 1977.

[3] Accelerator Control Systems, Proceedings of the Second International Workshop on Accelerator Control Systems, Nuclear Instruments and Methods, Vol. A247, No. 1, 1985.

[4] Proceedings, Europhysics Conference on Control Systems for Experimental Physics, CERN Report 90–08, 1990.

[5] N.D. Arnold, G.J. Nawrocki, R.T. Daly, M.R. Kraimer, W. P. McDowell, "I/O Subnets for the APS Control System," to be published in the Proceedings of the 1991 Particle Accelerator Conference, San Francisco, California, May 6–9, 1991.

[6] J. O. Hill, "Channel Access: A Software Bus for the LAACS," in Accelerator and Large Experimental Physics Control Systems, D. P. Gurd and M. Crowley–Milling, Eds., ICLAEPCS, Vancouver, British Columbia, Canada, pp. 288–291, 1989.

[7] Accelerator Automation Application Toolkit Workshop, Sponsored by AT Division and MP Division, Los Alamos National Laboratory and CERN, Geneva, 31 October – 4 November, 1988.

[8] L. R. Dalesio, M. R. Kraimer, A. J. Kozubal, "EPICS Architecture," to be presented at ICALEPCS '91, Tsukuba–shi, Ibaraki, Japan, November, 1991.

[9] R. T. Daly, F. C. Vong, M. R. Kraimer, "The APS Radio Frequency Test Stand Control System," presented at Real Time '91, Computer Applications in Nuclear Particle and Plasma Physics, Julich, Germany, 25–28 June, 1991.

[10] M. R. Kraimer, B. K. Cha, M. D. Anderson, "Alarm Handler for the Advanced Photon Source Control System," to be published in the Proceedings of the 1991 Particle Accelerator Conference, San Francisco, California, May 6–9, 1991.

The ESRF Control System; Status and Highlights

W.-D.Klotz
European Synchrotron Radiation Facility
BP 220
38043 Grenoble Cedex, France

1 Introduction

The European Synchrotron Radiation Facility[1][1] will operate a $6GeV$ e^-/e^+ storage ring of 850m circumference to deliver to date unprecedented high brilliance X-rays to the European research community. The ESRF is the first member of a *new generation* of Synchrotron Radiation Sources, in which the *brilliance* of the beam and the utilization of *insertion devices* are pushed to their present limits.

Commissioning of the facility's storage ring will start in spring 1992. A full energy injector, consisting of a 200MeV linear preinjector and a $6GeV$ fast cycling synchrotron ($10Hz$) of 350m circumference have been successfully commissioned during the last months.

The machine control system for this facility, which is under construction since 1988, is still under development, but its initial on-site operation this year has clearly made easier the commissioning of the preinjector plant.

A description of the current system is given and application software for start-up is briefly described.

2 Architecture

The ESRF control system is based on a multi-level architecture of distributed hard- and software processing units[2]. Logically the system is structured into four levels. From top to bottom we call them:

- Console Level (Presentation);
- Process Level (Applications);
- Group Level (Device Servers);
- Field Level (Equipments, Embedded Controlers).

On the lowest level, all equipments are interfaced; either by intelligent controlers, as they are delivered from the manufacturer, or by dumb interfaces. Equipments are logically grouped together on the group level. Grouping of equipments is done by similar functionality. The group level is responsible for hardware specific and real time I/O-operations. *Device servers* perform the task of hiding hardware specifics to the upper level. The process level represents that level of the control system where practically all higher level control tasks take place and where physics applications are processed. Powerful multitasking capabilities and fast processing is mandatory on this level. The presentation level presents the interface between the operators and the system. Within this level data entering from the lower level are presented graphically or are formatted to readable reports. Commands entered by means of interactive devices are decoded into events and finally passed as internal messages to the lower levels.

Physically the system is split into 2 levels. All nodes of the presentation and process level consist of UNIX based workstations and file/compute servers interconnected by Ethernet. The group level nodes are realised by VMEbus crates, equipped with 68030 CPU boards. These systems run the OS9 multitasking real time kernel/operating system. Every process level server connects to a private Ethernet segment onto which group level nodes it is in charge of are connected.

The physical boder line between group level and field level is fuzzy. In our system some dumb devices are directly interfaced to VME I/O-boards that are plugged into group level crates, but most dumb devices are interfaced by means of G64 crates. Groups of G64 crates, that interface classes of similar devices, are connected to multidrop highways that are mastered by group level crates. This multidrop highway[2] was developed at ESRF. However, intelligent devices with embedded controlers are, in the majority of cases, directly connected to VME (group level) crates[3].

3 Networks

Modern control systems are distributed[3, 4]. The larger the accelerator is, the more important is its network infrastructure. The ESRF control system is fully distributed and relies strongly on a high speed computer network.

Figure 1 gives an overview of the logical and physical implementation of the control system and its network.

Apart from the home-made multidrop highway all computer connections are based on the Ethernet(IEEE 802.3)

---

[1] (ESRF)

[2] we named it FBUS

[3] in the majority by RS422 or RS232 asynchronous serial links

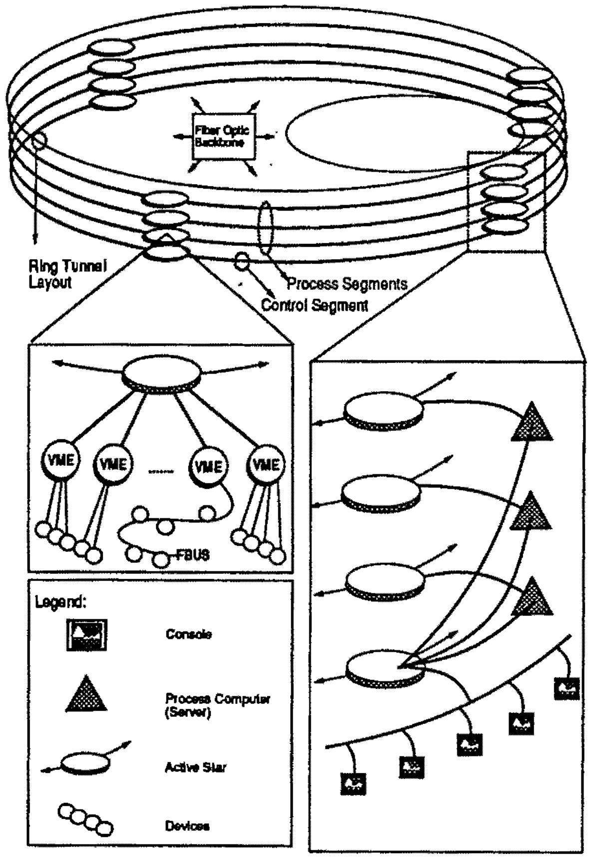


Figure 1: Overall Layout of Control System and Networks

LAN standard and the TCP/IP protocol suite. The network is constructed using $50/125\mu m$ multimode graded index optic fibres[4] for cabling, that will allow a later migration to FDDI.

Four networking centers are located around the storage ring tunnel, a center comprising a "NODE" and a wiring "HUB". A NODE is the location that acts as the convergent point for IEEE 802.3 compliant active devices e.g. an *active star*. To the NODE are attached "LOBES" which are configured on a star wired topology. A LOBE is the remote system connected to a NODE, and in this case it is a remote "transceiver" attached by a standard AUI[5] cable to a VME crate, process computer, graphics workstation or fanout unit.

For the active stars at the NODES, the "Lannet Multinet II" system was chosen for its ability to operate in a single chassis and support up to four independent backbone bus's. In addition all backbone fibre optic links are run in a *synchronous* Ethernet mode. By using this mode it is possible to install more than the "four" repeaters that restricts standard asynchronous Ethernet systems, with the restraining factor being, the round trip delay that limits each Ethernet segment to approx. 4.5 kilometers.

The wiring HUB is the central point for passive network components, i.e. the *backbone optical fibres* that link all the HUBs together in a circular structure around the storage ring tunnel. It also acts as the termination point for all star wired fibres that are attached to the LOBES.

When installing the circular *backbone*, provision for 10 independent rings has been made, out of which four will be used initially. One ring functions as the main *control segment*, to which all upper layer processors are connected. There are three dedicated *process servers* that operate as network gateways to the other three rings. The latter are used as *process segments* to which all middle layer VME systems are connected. Devices in the field are either connected directly to them or accessed through the multidrop highway.

Response times on a heavily loaded Ethernet become quasi stochastic. Having this in mind, the whole system was designed in a way to provide maximum flexibility in distributing sinks and sources of network traffic. This is accomplished by fiber optic patch panels situated by the HUBs. By simply crosspatching between panels, any processor can immediately be connected to any of the independent rings comprising the backbone. Upgrading the backbone to FDDI would be another, although much more elaborate, remedy against network congestion.

4 Computers

4.1 Consoles

The standard console in the control room will be an HP Apollo 9000 Model 720 workstation with local disk to hold the bootable image of the operating system and to provide local swap space. File systems containing control system and physics applications software are remotely mounted[6] from the process servers. Currently we are running 5 consoles in the main control room.

In addition to that, RISC based X Window graphics terminals, like the HP 700/X familiy, that are served by the process servers, will be used as console devices. Their ideal usage will be that of "remote" consoles. They will be outside of the control room but plugging to the backbone's control segment. Their main purpose is to give scientists, working on an experiment, the possibility to control the movements of *their* "insertion device" by themselves.

4.2 Process Servers

As process servers the HP 9000 Series 800 Model 842 and 857 midrange super-mini computers were selected. There are currently three of these machines in the system. System features are:

- high reliability;

---

[4] Bandwidth specified at $850nm > 500MHz * km$ and at $1300nm > 700MHz * km$

[5] Attachment Unit Interface

[6] currently we are using SUN's NFS that we hope to replace by OSF's DCE

- 29 Mips; 6.9 DP[7] Mflops; CMOS RISC technology;
- internal disk storage: <2.68 Gbytes;
- I/O bandwidth 21 Mbytes/sec;
- integrated DAT[8] unit: 1.3 Gbytes capacity;

These machines are configured as network *gateways* between the control segment and one of the corresponding process segments. To accomplish this, each of them is equipped with two high speed LAN adapters.

The process servers and the console workstations run HP-UX; an AT&T System V Rel 3.0, and BSD 4.3 compliant implementation of the UNIX operating system. All machines run MIT's X Window System, which allows applications to run as X-*clients* on the process servers and to perform interactive I/O through X-*servers* running on the consoles.

4.3 VME Systems

All VME systems have an identical base configuration. This comprises a CPU with Motorola 68030 @ 20 MHz, 4 Mbyte RAM, on board Ethernet adapter, and additional 512Kbyte battery backed up RAM on a separate board.

On these systems we run Microware's OS9 realtime kernel/operating system. In addition the systems are equipped with a TCP/IP Internet Support Package providing Berkely sockets, and SUN's Network File System.

All systems are running in a *diskless* configuration for ease of maintenance and reliability. The battery backed 512Kbyte RAM holds the OS9 real time kernel and a minimal TCP/IP configuration. When the system boots, it loads additional OS9 modules, applications, and the NFS modules from a process server. Using NFS, it then copies configuration files from the process server into its RAM disc and initialises the applications.[9] Cold start-up still has to be done manually by toggling a switch on the board's front panel, but we are working on a remote facility.

5 Interfacing

Many of the VME systems drive fast multidrop highways. This FBUS is not a general purpose network but implements a low cost remote input/ouput facility. It relies on a master-slave relationship, where a controller (VME based module) drives a large number[10] of slave nodes. The nodes comply with the G64 standard, so that full advantage can be taken of existing interface boards from industry. The FBUS multidrop highway is based on a synchronous serial protocol. Physical implementation uses an extremly noise resistant Manchester encoding with transformer isolation, i.e. each node being galvanically isolated from the highway. The data rate is up to 2 Mbits per second on a 200m-300m long highway. FBUS can still be safely operated at a speed of 1 Mbits per second on distances of up to 1 km and 30 nodes without repeater. The transmission medium is a flexible shielded twisted-pair cable with a characteristic impedance of 78 Ohms.

G64 and FBUS interfacing has been used for control of main magnet power supplies, beam position monitors, magnet interlocks, corrector magnet power supplies, and injection/extraction elements. Other significant subsystems that include G64 crates are the system to distribute the slow timing pulses, the video cross point switch, and the video multiplexors for fluorescent screen monitors. The rest of devices is directly interfaced to the VME systems; either by asynchronous serial lines or digital I/Os.

As a result of an early taken policy, to stick to industry standards, *only a few* boards had to be designed by the ESRF Digital Electronics team:

- video multiplexor[11] (VME)
- delay unit[12] (VME)
- ADC with on-board memory (G64)
- FBUS master (VME and PC/ATbus)
- FBUS slave (G64)
- clock divider[13]

Unavoidably some other dedicated electronics had to be designed to adapt some exotic devices to the standards chosen.

Table 1 gives an overview of interfacing.

6 Software

6.1 Equipment Access

The low level system software for the distributed control system is based on a "Client/Server" architecture. The Client/Server technology is a simple mechanism to distribute software tasks across any number of processors. This approach is *open and object oriented*, can be implemented on existing systems (eg. OS9 and UNIX), and will be discussed in detail by a contribution to this conference from A.Götz.

Objects are sets of *hidden data* on which well *defined operations* may be performed by authorized users. Associated which each object is a "Server" process that manages the object and exports its functionality as a service. This server-based model is implemented using "Remote Procedure Calls"[14][5, 6]. When a user process wants to perform an operation on an object, it sends a request message to the Server in charge of it. The message contains access keys, a specification of the operation to be performed, and any parameters the operation requires. The user process,

---

[7] With floating-point processor, Double Precision

[8] Digital Audio Tape

[9] Microware has recently started to ship the BOOTP Port Packs for OS9. BOOTP is used to make a boot PROM with the possibility to boot OS9 directly over Ethernet, thus avoiding the battery backed up RAM

[10] up to 64 on one highway

[11] 15:1

[12] 6 channels, 32MHz resolution, 0-524 ms range

[13] for RF-synchronous triggers, 352MHz:32MHz

[14] RPC; in our implementation we use SUN's RPC and XDR

VME crate	**VME** dig I/O	**VME** serial I/O	**VME** special I/O	**G64** crate	**G64** analog in	**G64** analog out	**Where**
1	20	8	6	15	84	84	Transfer Line 1
1	48		6	4	3	3	Transfer Line 2
1		6	2	3			Injection/Extraction
1			16	5	75		Synchr. Diagnostic
2		3	15	9			Synchr. Magnet Power Supplies
3		56					Synchr. Vacuum
1		34	36	8			SY/SR slow timing
1			37	33	224		Storage Ring Diagnostics
1		21	1	36	36	36	Storage Ring Magnet Power Supplies
16		448	34	32	2050		Storage Ring Vacuum
1		1	5	51	576		Geodesy & Mag. Interlocks
14		48	14	9		9	Insertion Devices
			68	17	816		Front Ends
		41	14		82		X-ray BPMs
2	34	53					Radiation & Safety
2				2			misc. Monitoring
4							RF & LINAC (subcontr.)
51	102	719	254	224	3946	132	**TOTAL**

Table 1: Overview of Hardware Interfacing for the ESRF accelerator plant

known as the "Client", then blocks. After the Server has performed the operation, it sends back a reply message that unblocks the Client. The combination of sending a request message, blocking, and accepting a reply message forms a Remote Procedure Call, which can be encapsulated to make the entire remote operation look like a local procedure call.

The lowest level of objects in the accelerator plant are the actors and sensors (or physical devices). These objects are "terminator objects"; they are easy to identify, and their behaviour can be modelled and documented.

"Device Servers" are terminator objects that operate on physical devices. The more general term "Server" or "Virtual Device Server" covers objects on a higher level of abstraction. Objects on a higher level of abstraction can use terminator objects to offer a given service[15].

The Device Server is an intermediary between application programs and the physical resources of the accelerator system. It contains all device-specific code, and insulates applications from differences between hardware. It performs the following tasks:

- Allows access to the device by multiple clients. To implement security, the server, depending on its state, may deny access from certain clients.

- Interprets network messages from clients and acts on them. Messages are generated by clients through RPCs.

- Maintains complex data structures, including device state information. Server maintained information reduce the amount of data that has to be maintained by each client and the amount of data that has to be passed over the network.

At ESRF the Device Servers follow the OOP[16] paradigm and have to be implemented in a certain, fixed style. The OOP paradigm is based on the "widget" model from the X11 Intrinsics Toolkit[7] of MIT and is implemented in ANSI C.

The control system designers have decided to implement *all* important functions necessary to run the distributed system in Servers. This includes the processes to boot and manage the system, to access the database, to handle graphics objects, as well as Device Servers to access equipments.

According to our current state of knowledge about 53 Device Servers have to be written for the complete system. About 50% of them are currently released. The average size of a Device Server ranges typically between 2000–2500 lines of C code.

---

[15] The encapsulation of lower level services into higher level services can continue until a very high abstraction like physical machine parameters, i.e. energy, chromaticity, tune, emittance,..., is achieved.

[16] Object oriented programming

6.2 Graphics User Interface

This field of technology is in a state of tremendous innovations. Fortunately some standards exist now: **X11**, and **Motif**. The X11-window system, or X11, is a network-transparent window system. With X11, multiple applications can run simultaneously in windows, generating text and graphics. Network transparency means that application programs that are running on other machines scattered throughout the network, can be used as if they were running on a local machine.

The core components of the OSF/Motif technology include an extensible user interface[17], an applications programming interface[18], a user interface metalanguage[19], and a window manager. Motif is based on the X Intrinsics, a toolkit framework provided with X11. The Intrinsics use an object oriented model to create a class hierarchy of graphical objects known as "*widgets*".

Both X11 and Motif, are extremely helpful but their libraries are complex to learn and to use for programming. Coding of applications started initially with those libraries, and demanded substantial efforts in becoming familiar with this new technology.

User Interface Management Systems[20] (sometimes called interface builders) are the tools which help the application programmer to design the user interface part of the application interactively. A UIMS is generally composed of a graphic oriented editor and a code generator, and sometimes other complementary tools. There are several Motif compliant UIMS available. A few of them have been tested at ESRF. At present one of them has been selected for use[8].

This UIMS drastically eases now the design of Motif-based user interfaces. It generates stand-alone C code and/or a combination of Motif-compliant C and UIL code.

Synoptic drawings with selectable objects are scarcely supported by the above mentioned tools. We therefore work on an implementation of a Motif compliant widget that uses vectorial drawings generated by PHIGS[21]. Synoptics will be generated by CAE systems like AUTOCAD or EUCLID.

6.3 Database

The control system data are stored in relational databases which manage two logical parts:

- Resource data;
- Runtime data.

The resource database keeps permanent data. Examples are: start-up resources, calibrations, equipment definitions, installation- & maintenance data, etc...

The implementation of the resource database uses ORACLE and its powerful set of development tools. Modifications on the resource database cause automatic update of runtime data sets, that are redundant copies of the *parent data set* in the resource database.

The runtime database is a central warehouse for all sorts of temporary or transient data. It is not a medium for permanent storage. It is simply a front for permanent storage. Only *memory resident* database systems can meet the demands for sufficiently short transaction times. A prototype of the runtime database is operational and uses a *Real-Time Database Base Management System*[22] available on HPs.

The runtime database can be used to alleviate congestion problems. Multiple processes can update data asynchronously in the database. Other processes can retrieve this information aynchronously without blocking the process doing the updating. Since the memory resident database profits from a much higher than normal I/O throughput, this mechanism is used to resolve information traffic jams that may occur.

The runtime database's prime source is a so-called *Update Daemon* that updates the current machine status if a particular client requests this. The database contains ring buffered tables to store brief histories of the results of requests issued to a device. Although physically the runtime database is distributed over all process servers, access to it is transparent to database clients. On-line data can be archived continously. Only a time window of some five minutes is kept in memory by RTDB, the rest of the data is dumped into the disk-based ORACLE database. An index to these data is constructed to allow *queries* in accelerator physics terms on archived data. Data can be stamped with time or accelerator status information.

The same runtime database can also be used by applications as a mean for interprocess communication. Applications dynamically allocate "tables" of formatted data, that can then be piped or multiplexed to other applications.

The volume of data that will be managed by the resource database is estimated to be some 10Mbytes. The whole control system comprises more than 3000 devices and more than 50000 static resources. The throughput of on-line data at its worst is estimated to be some 40kbytes/sec. If all data coming from the machine is stored at an interval of a second, it would mean 12Mbytes every 5 minutes.

6.4 Applications

Application program development at ESRF has been taken care of at an early stage of the project. Usually this class of software tends to be too little and too late. To give accelerator physicists the possibility to develop their applications in parallel with the control system software, an applications programmer interface[23] has been defined very early and kept stable until then.

---

[17] UI
[18] API
[19] UIL
[20] UIMS
[21] stands for Programmer's Hierarchical Graphics System and is an ISO standard
[22] called RTDB
[23] API

	NFS/RPC		API
	UDP	TCP	UDP
open connection	20-25ms	35-45ms	55-65ms
close connection	0.1-0.2ms	0.3-0.5ms	10-20ms
RPC with 100bytes	10-15ms	15-20ms	15-20ms
RPC with 8kbytes	25-35ms	55-65ms	30-40ms
RPC with 40kbytes		220-250ms	

Table 2: Performance figures for RPC and API

Access to the Device Servers is provided by a small set of C calls. These calls allow the users to develop their applications in peace without being affected by what goes on in the Device Server software. Initially very simple Device Servers have been written, that ran on the local host, and that only *simulated* devices. These API-calls hide the complexity of Device Servers and their implementation from users by offering them a set of high level commands as access method. *How* and *where* the Device Server executes the high level command is hidden. In the distributed environment this workload is spread over a number of machines.

The following functions form the basis of the Device Server API:

- `int dev_import(name, access, ds_ptr, error)`

  ```
  char *name;          /* Device Server Name */
  long int access;     /* Access Type */
  devserver *ds_ptr;   /* DevServer Handle */
  long *error;         /* Error Code */
  ```

  Is called by the application to establish a connection to a Device of the specified name.

- `int dev_putget(ds, cmd, argin_ptr, typein argout_ptr, typeout, error)`

  ```
  int dev_putget(ds, cmd, argin_ptr, typein
                 argout_ptr, typeout, error)
  devserver ds;          /* DevServer Handle */
  DevCmd cmd;            /* Device Command */
  DevArgPtr argin_ptr;   /* Call Parameter */
  DevType typein;        /* Parameter Type */
  DevArgPtr argout_ptr;/* Return Parameter */
  DevType typeout;       /* Parameter Type */
  long *error;
  ```

  Is called by the application to execute a command on the device. This is a "blocking" call which doesn't return until the command requested has been executed.

- `int dev_put(ds, cmd, argin_ptr, typein, error)`

  Is called by the application to execute a command on the device. This is an "asynchronous" call which will return as soon as the command has been delivered to the server or an error occurred. This call can only be used to start a command, no knowledge is returned about its execution and/or success. It is up to the application to interrogate the Device Server to determine its status.

- `int dev_free(ds, error)`

  ```
  DevServer ds;    /* DevServer Handle */
  long *error;
  ```

  Is called by an application to release a device properly.

Measurements of RPC and API performance that we achieve between a process server and a VME node are given in table 2.

Using strictly this device access interface and the X11 and Motif standards for interactive graphical I/O, an impressive number of physics applications have been developed in parallel with the basic control system software, and have considerably helped to commission the booster in due time. An enumeration of applications presently available follows:

Transfer line 1 & 2: These programs execute specific procedures for step by step alignment, emittance measurement, and modelling of beam envelopes.

Closed orbit: The program performs basic control of steerers and bumps, beam position readout, orbit plots, fourier analysis of orbit and steerers, and automatic orbit correction.

Booster vacuum: This program controls the vacuum system. It allows individual device control, display of periodically updated status, and display of pressure profile.

Booster injection/extraction: This program allows control of current- and timing settings of pulsed injection/extraction elements.

Booster optics: Different options in this application allow tune measurement/setting, chromaticity measurement/setting, measurement of β-functions at quadrupole locations, and measurement of dispersion (η-function).

Storage ring injection: Used for tuning of the injection kicker/septa to maintain an injection bump and control injected beam position and angle.

7 Conclusion

The ESRF control system is operational since August 1991. It played an important role during commissioning of the booster synchrotron. The system has been designed from bottom up, using object oriented programming techniques, and is based on proven industry standards. Its design has been guided by a clear preference for mature commercial systems over custom- or home-made ones, without formally excluding the latter.

The system is not finished yet, but it is easily extendable and adaptable to future needs. It is through the standards that have been selected for the control system, that ESRF will be able to migrate together with industry to new technologies while preserving considerable investments in hard- and software. The choices of UNIX, X11/Motif, VME/OS9, Ethernet, and TCP/IP have been fundamental in this sense.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S03SRD05

8 Aknowledgements

The author is grateful to all contributing members of the ESRF controls group; in particular to C.Hervé, head of the Digital Design Group, and A.Götz, head of the Control Software Group, for their competent and efficient leadership of the project. He takes the opportunity to thank also A.Ropert, L.Farvacque, and all members of the machine division Theory Group, for their ongoing engagement in the production of control- and physics applications, and the smooth collaboration with them. Thanks also to J.Altaber, and F.Perriollat (both CERN), as well as collaborating groups from CERN, Trieste, and Jülich, who initially helped us to get the project going.

9 References

References

[1] J.L. Laclare, "Overview of the European Synchrotron Light Source" in *IEEE Particle Accelerator Conference*, Washington, D.C., March 1987, pp. 417-421.

[2] W.D. Klotz and C.Hervé, "The Conceptual design of the ESRF Control System" in *European Particle Accelerator Conference*, Rome, June 1988, pp. 1196-1198.

[3] *Proceedings of Europhysics Conference on Control Systems for Experimental Physics*, Villars-sur-Ollon, Switzerland, Sept.28 - Oct.2, 1987.

[4] *Proceedings of the International Conference on Accelerator and Large Physics Control Systems*, Vancouver, BC, October 30 - November 3, 1989.

[5] Pal S. Andersen, V. Frammery, and R. Wilcke, "Tools for Remote Computing in Accelerator control" in *Proceedings of the International Conference on Accelerator and Large Physics Control Systems*, Vancouver, BC, October 30 - November 3, 1989, pp. 225-230.

[6] *Network Programming Guide*, SUN microsystems, Part Number: 800-3850-10, Revision A of 27 March, 1990.

[7] A. Nye and T. O'Reilly, *X Toolkit Intrinsics Programming Manual*, O'Reilly & Associates, Inc., 1990.

[8] *The Builder Xcessory User Guide*, Cambridge, MA: Integrated Computer Solutions, Inc. 1990

Trademarks

UNIX is a trademark of AT&T in the USA and other countries.

OS9/68000 is a trademark of Microware Systems Corp., USA.

OSF/Motif is a trademark of the Open Software Foundation, Inc.

OSF/DCE is a trademark of the Open Software Foundation, Inc.

Builder Xcessory is a trademark of Integrated Computer Solutions, Inc.

NFS is a trademark of SUN Microsystems, Inc.

XDR is a trademark of SUN Microsystems, Inc.

Ethernet is a trademark of Xerox Corp.

The X Window System is a trademark of the M.I.T.

HP-UX is a trademark of Hewlett Packard, Corp.

ORACLE is a trademark of Oracle Corp.

RTDB is a trademark of Automated Technology Associates, Indianapolis, USA

AUTOCAD is a trademark of Autodesk, Inc.

EUCLID is a trademark of Matra Datavision, S.A.

All other trademarks or registered trademarks are of their respective companies. ESRF disclaims any responsability for specifying which marks are owned by which companies or organizations.

Centralized Multiprocessor Control System for the Frascati Storage Rings DAΦNE

Giampiero Di Pirro, Catia Milardi, Mario Serio, Alessandro Stecchi, Luciano Trasatti.
INFN, Laboratori Nazionali di Frascati
P. O. Box 13, 00044 Frascati (Rm) ITALY, e.m. trasatti@irmlnf
and
Barbara Caccia, Vittorio Dante, Raffaellino Lomoro, Eleuterio Spiriti, Stefano Valentini.
Istituto Superiore di Sanita' and INFN, Sezione Sanita',
Via Regina Elena, 299, 00161 Roma, ITALY, e.m. caccia@sanita.infn.it

Abstract

We describe the status of the DANTE (DAΦne New Tools Environment) control system for the new DAΦNE Φ-factory under construction at the Frascati National Laboratories. The system is based on a centralized communication architecture for simplicity and reliability. A central processor unit coordinates all communications between the consoles and the lower level distributed processing power, and continuously updates a central memory that contains the whole machine status. We have developed a system of VME Fiber Optic interfaces allowing very fast point to point communication between distant processors. Macintosh II personal computers are used as consoles. The lower levels are all built using the VME standard.

I. DAΦNE

DAΦNE [1] is a two ring colliding beam Φ-Factory under construction at the Frascati National Laboratories (See Fig. 1).

Construction and commissioning is scheduled for the end of 1995.

The luminosity target is ~10^{33} cm^{-2} sec^{-1}.

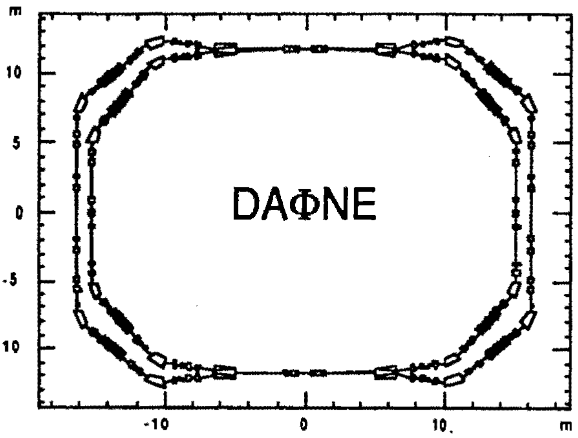


Fig. 1 : The DAΦNE Φ-Factory layout

II. SYSTEM STRUCTURE

Fig. 2 shows the general architecture of the control system. Three levels are defined:

PARADISE (PARAllel DISplay Environment) is the top level, implementing the human interface. Several consoles, built on Macintosh personal computers, communicate with the rest of the system through high speed DMA busses and fiber optic links.

PURGATORY (Primary Unit for Readout and GATing Of Real time Yonder) is the second and central level of the system. It essentially contains only a CPU and a Memory in a VME crate. The CPU acts as a general concentrator and coordinator of messages throughout the system. The central Memory is continuously updated and represents the prototype of the machine database.

HELL (Hardware Environment at the Low Level) is the third level of the system and is constituted by many (about 60) VME crates distributed around the machines.

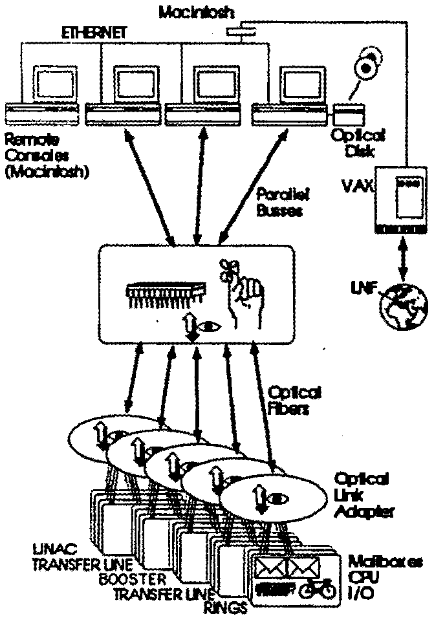


Fig. 2: Control System Schematic Diagram

A CPU in every crate performs control and information hiding from the upper levels.

VME is used throughout Purgatory and Hell

A first estimate of the system gives about 7000 channels to be controlled.

Centralized Communication Control

We have chosen an architecture based on a single central controller of communications instead of the usual network for reliability and performance.

A system with a central CPU controlling all the others through high speed point to point links and a polling mechanism is much easier to implement and to maintain than a network.

Data integrity is easily achieved through a write-readback mechanism and the failure of a peripheral unit can be diagnosed and isolated very efficiently.

Performances are much better than usual networks, due to link speed and protocol simplicity: in a previous paper[3] we measured:

1700 messages/s from Paradise to Hell
10 µs polling time for Purgatory to check out a Hell CPU
450 KByte/s data transfer from Hell to Paradise.

In this architecture the link between the consoles and the central coordinator (Purgatory) is easily implemented through high speed DMA busses. The same is not true for the links between Purgatory and Hell, since the peripheral CPUs are spread over a wide geographic area, well above the standard 70m allowed by DMA busses. The best solution for these links is to use fiber optic connections, with their high bandwidth and noise immunity.

III. OPLA' (OPTICAL LINK ADAPTER)

The OPLA' (OPtical Link Adapter) project for an interface between a standard third level VME crate, and an optical fiber has been developed. This project aims at realizing a multipurpose system for fast data transfer over long distance.

Use of the AMD Taxi chips allows data rates of up to 160 Mbit/s, which is more than a standard CPU can transmit on a VME bus.

A simple architecture will be implemented: a 16 bit word presented to the transmitter will be stored on the other side of the link in a 2048 word FIFO. FIFO overflow will be automatically prevented by back transmission of a FIFO-full status message.

A first prototype board has been developed and tested. The board can be divided into two sections:

i) Tx/Rx toward the optical fiber;
ii) VME interface.

Software for controlling and testing the board has been developed on a Macintosh IIfx using LabVIEW®[6]. LabVIEW® allows to create a front panel that specifies inputs and outputs providing the user interface for interactive operations. Behind the front panel there is a block diagram, which is the executable program.

A panel for the preliminary tests on the OPLA' prototype boards has been built This panel allows access to two VME boards connected through optical fibers.

To access to VMEbus a MICRON (Mac Vee Interface Card Resident On Nubus)[2], developed at CERN and a MacVee (Microcomputer Applied to the Control of VME Electronic Equipment) are used.

The next step of the project will be to implement four complete Tx/Rx sections on a single board. In fig.3 the schematic design of the board is reported. Gate arrays will be used to reduce component count and therefore design time and number of required boards. For each section a Xilinx programmable gate array will implement the control of data transmission and the receive logic. A single gate array will implements the interface toward VME.

The AMD Taxi chips are used to implement the interface toward the optical fiber, due to their ease of connection and high integration.

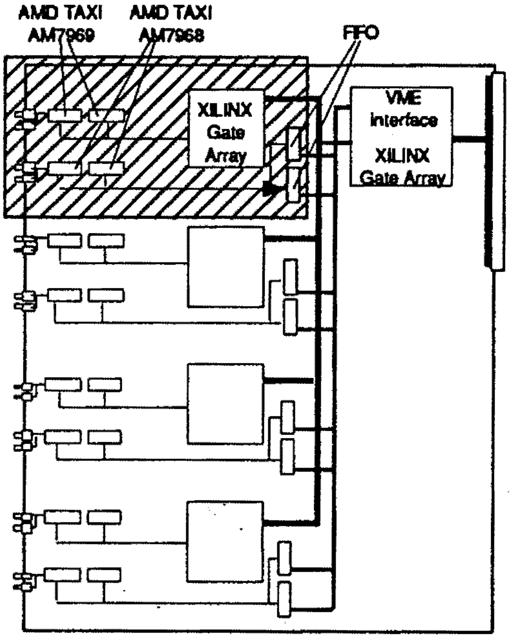


Fig. 3 : OPLA' schematic diagram

IV. CONSOLES

Macintosh personal computers have been chosen for the system consoles. In the last few years we have seen an impressive effort by personal computer firms and third parties to supply large quantities of very high quality software at very low prices. The situation is now, as far as software is concerned, definitely in favour of the use of large diffusion machines as opposed to high cost, "high" power, low diffusion workstations. Hardware prices keep getting lower, while the cost of software development has reached about 80% of the total cost of an installation, with all the reliability risks of in-house software development. The Macintosh family of computers is at the moment the best candidate for a human interface development, since the effort expanded on software development on this machine has been the most striking on the market.

Previous experience with Hypercard [4,5], on the other hand, has shown that high level software packages can decrease software development times by strong factors. Faster and more powerful human interface packages are coming out every day.

We already mentioned LabVIEW®: it is the first large diffusion software package specifically designed for data acquisition and controls. Its main features are:

- Very easy creation of "virtual instruments", i.e. human interface panels containing controls and displays, acting on the appropriate hardware interface;
- Graphic development of programs through an "icon" language.
- Large scientific library containing frequently used tools (histograms, fourier transforms, etc.).

We are convinced that the use of this kind of large diffusion packages will allow us to save a very large amount of time and effort, not only in program development, but mainly in debugging and maintenance.

Remote Consoles

A good example of the above is the problem of remote consoles. Using an Ethernet or LocalTalk link and a commercial program, Timbuktu® [7] it is possible to gain complete control of a remote Macintosh, under the protection of a system of passwords. This was a specific request for the DAΦNE control system. How long would it have taken to build and debug such a facility?

V. VME OPERATING SYSTEM

In our previous experience with a similarly structured control system we used no operating system for the lower level CPUs. Simple FORTRAN or C programs took care of the relatively easy tasks of a small and dedicated CPU that only has to perform a few simple tasks. The general idea is still:"A CPU for each task". While this is a rather extreme statement, we think that the software environment for the lower level CPUs must be kept as simple as possible, at the expense of increasing their number. On the other hand, the advantages of using a standard environment are obvious as far as bookkeeping and standardization are concerned. At the moment we are evaluating several Real Time Kernels and we plan to reach a decision by the middle of next year.

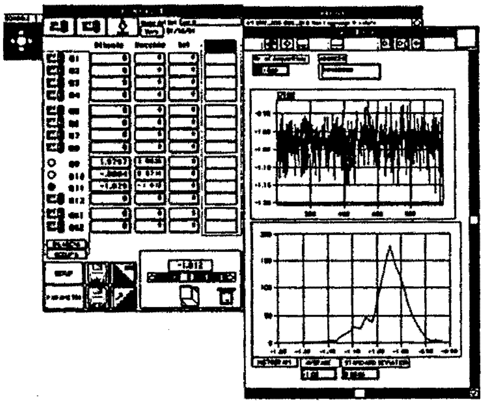


Fig. 4 : Field tests showing progressive migration from Hypercard to LabVIEW

VI. FIELD TESTS

We are testing out these ideas on small accelerators in Frascati. A first implementation on a set of steering coils at ADONE showed the feasibility of Hypercard as a human interface tool, at least for small systems.

Later, on a small machine, LISA, we have tried out the three level system and we have started migrating the human interface, originally written in Hypercard, to LabVIEW (see fig. 4). We have shown that a progressive migration is feasible, and these tests will allow us to measure the real performance of these software packages in the field. At the moment we believe that LabVIEW will prove adequate even for the large DAΦNE control system.

VII. SUMMARY

The control system we are building is based on highly distributed hardware and software capabilities, with a strong accent on openness to other environments. We believe that the human interface will be the most arduous problem to solve, and that the use of high diffusion software packages can be a big help in that direction.

A high speed fiber optic link adapted to the accelerator control environment is being developed.

VIII. ACKNOWLEDGEMENTS

We would like to thank the Accelerator Group of the LNF for continuing discussions and encouragement. The work with the LISA group helped us develop a set of techniques that will be very useful in the future.

IX. REFERENCES

[1] G. Vignola et al., talk presented at the San Francisco Particle Accelerator Conference, May 1991.

[2] B.G. Taylor, "The MacVEE Hardware User Manual"; Rev.4.1, CERN EP Division, 1987.

[3] L. Catani, G. Di Pirro, C. Milardi, L. Trasatti, F. Giacco, F. Serlini "The High Speed Bus System for LISA" Presented at the International Conference on "Accelerator and Large Experimental Physics Control System" Vancouver, Canada, Nov 3, (1989) *In Press to Nuclear Instruments and Methods.*

[4] L. Catani, G. Di Pirro, C. Milardi, A. Stecchi, L. Trasatti.:MARCO - Models of Accelerators and Rings to Commission and Operate, presented at the San Francisco Particle Accelerator Conference, May 1991.

[5] HyperCard v2.1, Apple Computer Inc, © 1987-90.

[6] LabVIEW, National Instruments Corp., © 1990.

[7] Timbuktu, Farallon Computing Inc, © 1987-91.

THE OPERATOR VIEW OF THE SUPERCONDUCTING CYCLOTRON AT LNS CATANIA

D.GIOVE, INFN MILAN
LASA - V. F.LLI CERVI 201
SEGRATE- ITALY
G. CUTTONE AND A. ROVELLI, INFN LNS
V.LE A.DORIA ANG. V. S. SOFIA
CATANIA-ITALY

Abstract

The upper level of a distributed control system designed for the Superconducting Cyclotron (SC), will be discussed. In particular, we will present a detailed description of the operator view of this accelerator along with the tools for I/O points management, data rappresentations, data archiving and retrieval. A dedicated program, developed by us, working under X-Window will be described as a starting point for a new man-machine interface approach in small laboratories opposed to the first industrial available packages.

I. INTRODUCTION

The SC developed at the Milan University, where was performed the first test on the magnetic field, has been moved to the final destination, at LNS in Catania. The work on the accelerator will start at the begin of the next year with the magnet excitation and the installation of the RF cavities. The extraction of the beam, injected in the booster by a 15 MV Tandem, is foreseen before the spring of 1993. The main features of this heavy ions accelerating facility are reported elsewhere [1] [2].

According to the experience gained in Milan on the control system during the first magnetic measurements, we planned improvements mainly on the upper level devoted to the man-machine interaction. The first console designed and realized in 1985-1987 [3], followed an old philosophy. Altough the main hardware and software choices had been proved satisfactory, it gave us a flexibility not so good as we expected.

Nowadays the availability of standard graphic software and the capability to create networks are the two features which make the workstation a pratical cost effective way to provide an universal environment for the development of the operator interface. It is possible to use powerful hardware and software standards which make straightforward the setting up of a network where hardware and software resources can be easily shared in a really efficient environment.

In the follow we will discuss the hardware and software architecture of the Superconducting Cyclotron operator console together with its performance measured during the test.

II. THE OPERATOR CONSOLE

In 1989, during the shut-down of the SC in Milan, we decided to review the structure of the console. Some general rules were fixed for the project.

- The architecure must be independent of the number of worksites in use: the insertion or the removal of a worksite must be invisible to the whole system, realizing in this way a real "easy expandible system".
- The architecture would allow to have the same graphical workstation conncted both in the main control room and in a remote place closed to the accelerator.
- The architecture of the console must be fully independent of the lower levels so that the choices made in process and plant levels don't influence the supervisor structure.
- The presentation level of the software must not require any practice in computer science and must be picture driven. The operator must be able to define its own working environment with few choices and the access to every information that he wants to deal with has to be guaranteed.
- The operations on an accelerator subsytem must be possible by each workstation but not at the same time. The display of all machine parameter must be possible on different workstations at the same time.
- The on line software configuration must be guaranteed by means dedicated programs taking advantage of a database.
- The allarms and malfunctioning logging task must be managed by a dedicated unit able to provide particular tools to help the operator in his trouble shooting job.
- The maintanance of the whole structure must be easy and centralized as much as possible on a single machine.

It was decided that the development of most of software would take not more than 3 man-years of work. The choice of the final solution was not easy and a lot of different considerations, like our experience with graphical workstations and their operating system, the estimated technical support avilable from vendors in our country, were taken into account. At last, we decided to implement our hardware architecture on a Local Area VaxCluster (LAVC) of 3100 Vaxstations with a µVax 3100 as boot member. A gateway was provided towards the lower levels. A µVax II was dedicated to this task along with the storage of the memory map of all sensors and actuators. Two 80386 PCs were dedicated to allarm logging and to manage the data necessary for the application tasks.

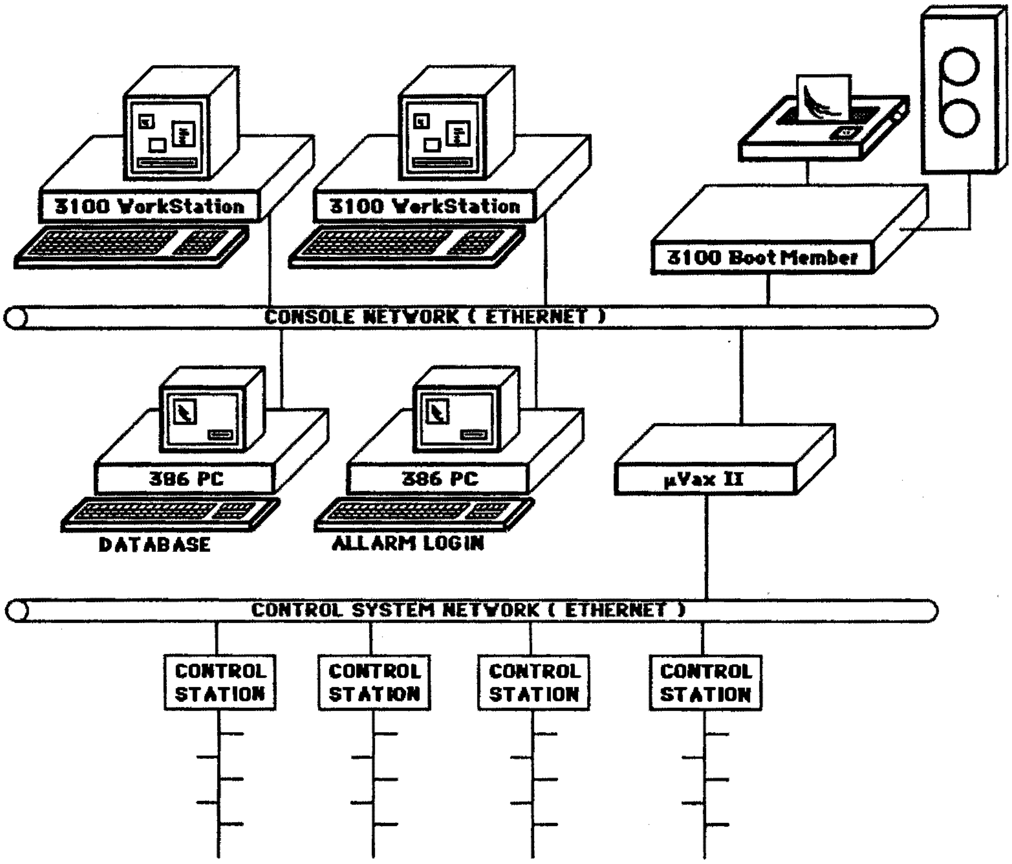


Fig. 1 The hardware architecture of the control system.

III. HARDWARE ARCHITECTURE

The main console has been designed as an Ethernet segment with a distributed software running on the resources which are connected to it. The Ethernet segment is physically the same which interconnects the process stations but we have superimposed on it the LAVC. This software is quite reliable and is well tested in a lot of installations all around the world. The structure resulting is easy to mantain as it recalls centralized architectures but it is really flexible and able to suit growing requirements. The intrinsic problem to have a central machine which will paralize the whole system in case of failure has been considered but our experience gained in similar architecture used for computer rooms reports a very low failure rate.

The boot member is dedicated to the management of Cluster operations and of print and tape resources. Historical logging of all the accelerator parameters acquired or calculated by the control system is provided by the µVax II, which is a satellite in the LAVC. Powerfull 3100 VaxStations with 16" Trinitron colour monitor have been chosen as the hardware platform for the operator interaction tools. All the workstations, whose number is completely unrelated to the architecture of the console, share the same programs and are able to work on every acces point to the coaxial Ethernet cable. This is the true sense of statement that: "the console is a network". A workstation will be dedicated to the beam dynamic simulation either in accelerator or in the beam transport lines. This machine, working on a complete accelerator or beam lines setting, will calculate the necessary corrections for the beam optimization.

Two particular nodes on the control room Ethernet are two 80386 based PCs, chosen because of their high performance, low cost, easy programming and full integration in the Digital network architecture (DECNET). The first is dedicated to run a database application (INFORMIX) used for the storage of all relevant parameters of each sensor and actuator of the cyclotron, like its name and software position in the control system, the different alarm thresholds and the range of possible setting value. Data are organized according to different query schemes to provide an interactive tool for everyone needs a particular information. A particular use of this application is the console itself. Infact, the data section of the applicative programs refers to this database and each variation is immediately available to the operator workstations taking advantage by the possibility in a DECNET environment to share disk between DOS and VMS machines. The second PC provides a lot of tools to help the operator during alarm handling and troubleshooting operations. This computer was conceived like an hypermedia machine, with enhanced graphic capabilities for the display of pictures or drawings related to particular event and with the possibility to play back defined speeches as an auxiliary tools. Fig. 1 showes the hardware architecture of the control room Ethernet so far described.

IV. SOFTWARE ARCHITECTURE

The most relevant challenge in the development of a new, flexible and portable console is in the software design. The whole hardware architecture described in the previous section would be meaningless if it would not be provided an adequate software support.

The main choice in the software architecture has been to use X-Window as basic development environment. Taking as a reference the software model proposed by X-Window, a modular object oriented code has been developed: GIULIA (Graphical Interactive Unit Level for Improved Automation). It is based on DEC-Windows, an extension of X-Window developed by DEC, which permits to use a powerfull toolkit and a User Interface Language (UIL) able to define the structure of the graphical interface. The Object description is stored in a separate file where the methods to which the objects will respond are outlined. The necessary code for the implementation of every method described in the UIL file is contained in a second file. Such a structure enhances the seperation between form and content of an application. The possibility to use low level Xlib functions together with DEC calls in the same code is guaranteed. We chosed to work under VMS and to use the "C" language to develop our code.

GIULIA is composed by two main parts running on at least two different computers: the first provides man-machine interaction tools while the second is dedicated to the management of the accelerator data received from the control stations and guarantees the operations on an accelerator subsytem by only a workstation at a time. GIULIA creates a remote task on the µVAX II devoted to the data exchange with the control system, in order to realize a trasparent task to task communication scheme. Data exchange has been optimized to reduce the traffic on the network and to semplify the handling of the shared memory. For each control station devoted to a particular accelerator subsystem according to a functional scheme of intelligence distribution, we have on the µVAX II two different VMS global sections where are written respectively the accelerator data received and the command to be sent. The access to these sections is controlled by means of flags. Tasks running on remote workstations look and actuate on the control hardware in an indirect way by means of this sections, making the software really device independent. Full operations on each accelerator subsystem are possible by each workstation asking to the µVAX II the allocation of this resource. At the end the operator must deallocate the resource to permit the operation by an other machine. The bandwidth measured on the network channel is of nearly 1.1 Mbit/sec for 2.5 Kbytes packets quite in agreement with the performance of other control networks. In this way we can send by a workstation up to 35 commands for second, receiving any time by the control station all the data acquired at that moment.

The interaction tools in GIULIA are based on the defintion of three different classes: rappresentation, interaction and BPM. Main attributes of these classes are "true" resizing and iconic interface. Rappresentation has three subclasses (table, bar chart and graphic) that identify the different ways to display every parameter of the cyclotron. Interaction has three subclasses (button, slider and knob) that identify the different ways to operate on each parameter of the cyclotron. A particular class is BPM which is realized to display the reconstructed beam shape acquired by the beam diagnostic devices, with a low frequency. The environment provided to the operator for its job is like that one of a writing desk: foils, particular rappresentations, are distributed on the screen and the parameters to be displayed or actuated have been chosen during a short navigation inside the program by the operator itself according to its preference. Foils may be opened, iconified, deleted and reconfigured. Graphic attributes of the subclasses may be easily changed by interactive setup utilities or modifing the UIL description file.

A process in GIULIA checks all variables for allarms or safety limits, and when a threshold is exceeded all the related informations are transferred to the second PC for further dedicated analysis. GIULIA provides an extensive on-line context sensitive help explaining how it works and all the details concerning the objects classes which it implements. Some choices, which may be dangerous as the modification of an allarm limit by the operator, are always recorded in the display of the workstation and are effective only for that related display.

V. CONCLUSION

The goal to develop the software so far discussed has been reached. Preliminary test has been carried out in order to verify the overall architecture and its performance that seems to be quite in agreement with the accelerator requirements.

In order to implement a real time picture of the beam a dedicated system is under development based on a CCD camera and a VME frame grabber board. A dedicated monitor will display the images so acquired at a high rate (about 30 Hz) with the possibilty to show up to 4 different images at the same time.

VI. REFERENCES

[1] E.Acerbi et al. "Progress Report on the Heavy Ion Facility at LNS" in EPAC Proceedings, Nice, France, June 1990, pp. 425-427.

[2] L.Calabretta et al. " The Tandem Injector for the Superconducting Cyclotron at LNS" in EPAC Proceedings, Nice, France, June 1990, pp. 497-499

[3] F. Aghion et al. "The Operator Interface in the Control System for the Milan Superconducting Cyclotron" in 11th Int. Conf. on Cylotrons and their Appl. Proceedings, Tokyo, Japan, 1987, pp. 432-435

The UNK Control System

V.N.Alferov, V.L.Brook, A.F.Dunaitsev, S.G.Goloborodko, S.V.Izgarshev, V.V.Komarov, A.F.Lukyantsev,
M.S.Mikheev, N.N.Trofimov, V.P.Sakharov, E.D.Scherbakov, A.I.Vaguine, V.P.Voevodin, V.D.Yourpalov,
S.A.Zelepoukin, IHEP, Protvino, USSR, B.Kuiper, CERN, Geneva

Abstract

The IHEP proton Accelerating and Storage Complex (UNK) includes in its first stage a 400 GeV conventional and a 3000 GeV superconducting ring placed in the same underground tunnel of 20.7 km circumference. The beam will be injected into UNK from the existing 70 GeV accelerator U-70. The experimental programme which is planned to start in 1995, will include 3000 GeV fixed target and 400*3000 GeV colliding beams physics. The size and complexity of the UNK dictate a distributed multiprocesssor architecture of the control system. About 4000 of 8/16 bit controllers, directly attached to the UNK equipment will perform low level control and data acquisition tasks. The equipment controllers will be connected via the MIL-1553 field bus to VME based 32-bit front end computers. The TCP/IP network will interconnect front end computers in the UNK equipment buildings with UNIX workstations and servers in the Main Control Room. The report presents the general architecture and current status of the UNK control.

1.Introduction

The UNK complex will combine - in one tunnel of 20.7 km circumference - a 400 GeV conventional magnet synchrotron (UNK-I) and a 3000 GeV superconducting synchrotron/storage ring (UNK-II). At a later stage a second superconducting ring (UNK-III) may be added with the aim of doing proton-proton collider physics at 6 TeV (Figure 1).

The UNK-I is injected at 70 GeV from the existing proton synchrotron U-70. For one filling up to 12 pulses from U-70 may be stacked, accelerated to 400 GeV and transferred to the UNK-II which in turn accelerates them up to 3 TeV.

Three main modes of operation are presently foreseen.

1. Fixed Target at 3 TeV: fast or slow extraction will send the 3 TeV beam to the fixed target experimental area. During the acceleration in the superconducting ring, U-70 may produce beams for its own 70 GeV experimental area.
2. Colliding Beams at 3 + 0.4 TeV: the beams from the UNK-II and UNK-I are made to collide. For this the UNK-I is operated first as booster and, after field reversal, as a storage ring run at 400 GeV.
3. Colliding Beams at 3 + 3 TeV: the UNK-I will first inject into one superconducting ring and, after field reversal, into the second one.

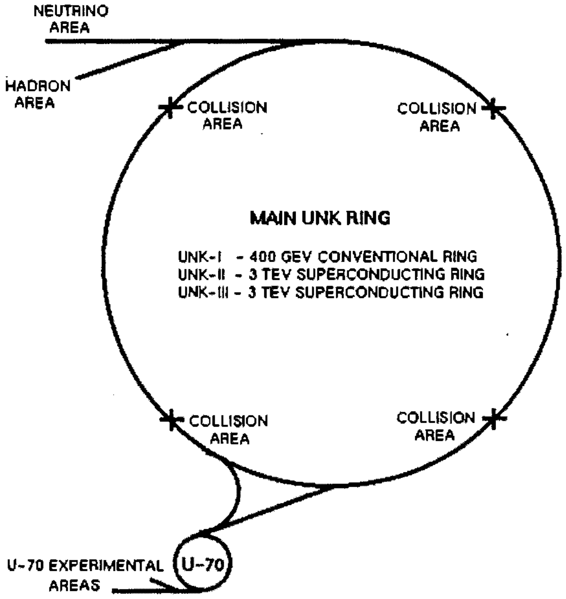


Figure 1 Layout of the UNK Complex

The accelerator controls equipment will be distributed over 24 on-surface buildings situated mainly along the accelerator ring. In one of the these is the Main Control Room (MCR), the other ones house the remote nodes of the control system. The latter are totally controlled from the MCR and are in general not manned. The typical distance between any two adjacent buildings is about 1.8 km and the maximum is about 3.5 km.

The more than 3500 superconducting magnets require a cryogenic plant and elaborate distribution, recovery and safety installations and their concomitant controls in the surface buildings around the ring tunnel.

The UNK operation is supported by general electricity and water distribution networks, tunnel ventilation, radiation protection, fire safety and other utilities which require highly reliable controls with 24 h/day 365 d/year availability.

The secondary beamlines and external experimental areas cover an area of roughly 12 km length. Controls for their

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S03SRD08

equipment may follow closely the principles of the accelerator controls.

The upgrade of U-70, for meeting UNK injector specifications, requires intensive machine studies which in turn make a controls upgrade mandatory. Since this must precede UNK, the principles and equipment may differ somewhat from UNK controls proper.

2. HARDWARE ARCHITECTURE

The size and complexity of UNK, together with real time and other requirements, dictate a multilevel multiprocessor controls architecture (see figure 2).

Various components of the accelerator equipment are driven by more than 4000 equipment controllers (EC) which perform low level control and data acquisition tasks in hard real time and provide a uniform equipment representation for the upper levels of the control system.

A typical EC is Euromechanics crate with Multibus I compatible backplane bus, contains a single board microcomputer, a fieldbus interface and a number of equipment specific I/O cards. The standard EC microcomputer is based on microprocessors similar to the Intel 8086/8087. In the UNK-I control system about 800 of such ECs will be used for beam instrumentation, power suplies, RF and vacuum system controls.

Similar technology will be used in the UNK-II and cryogenics complex equipment interface. The main difference is in the EC microcomputer type which in this case is based on the LSI-11 compatible microprocessors (see chapter 4). There will be about 1300 of such ECs in the UNK-II and cryogenics complex controls.

About 1500 of the UNK-I correction magnet power supplies will be controlled by embedded ECs based on the Intel 8051 microcontrollers. The EC is implemented on two standard Eurocards and performs all the power supply control functions, including timing and function generation. It has direct interfaces to the MIL-1553 fieldbus, timing and fast alarm systems.

A general timing system distributes reference events and clock trains to all ECs. A separate alarm and interlock network collects signals from all ECs monitoring vital accelerator subsystems. These signals may be used to trigger the beam abort system and inhibit beam injection.

The next higher level of control is represented by the front end computers (FEC) spread around the main UNK ring and beam transfer lines and interconnected with the upper level computers by the UNK controls network. The FECs will drive EC clusters through the 1 Mbit/s MIL-STD-1553 serial multidrop bus, which provides a cheap solution on a standard chip, noise immunity and galvanic insulation. Physically, the FEC is a modular board assembly in a standard VMEbus crate. The basic FEC configuration will consist of a 32-bit processor board, a network interface and a number of MIL-1553 bus controllers. It may also include I/O modules for direct interfacing to equipment needing full network functionality or/and bandwidth to handle high data rates (e.g. for sophisticated beam diagnostic devices).

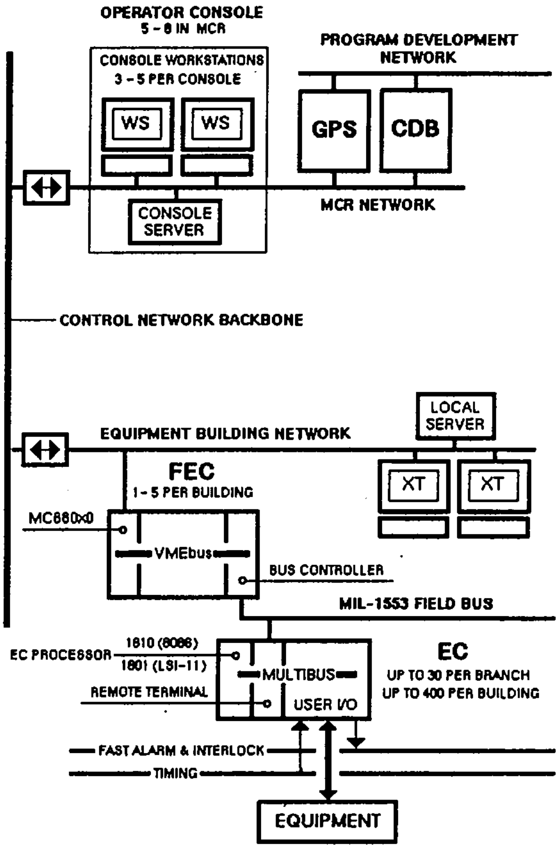


Figure 2 General Hardware Architecture

The FEC's main purpose is providing access for the upper level control software via network to the ECs. It may thus be considered as a gateway linking the MIL-1553 field bus to the UNK controls network and as a specific kind of network server, providing a set of equipment access services for application tasks running in the networked computers.

In addition to this general task, a number of FECs may be dedicated to certain functions through their own set of I/O modules. For example, one presently considers dedicating certain FECs to a task of UNK-II superconducting magnet main power supplies control and quench protection. Finally, a FEC may run some application tasks, in particular for local closed loop control and local equipment access, test and diagnostics. It will also perform ECs downloading and surveillance via the MIL-1553 field bus.

The network will be layered: a so-called backbone will interconnect the buildings and a number of LANs will interconnect equipment at the MCR and inside other buildings. The development infrastructure forms a sub-network which will be attached to the backbone. LANs will mostly be Ethernet. The backbone will be fiber optics FDDI or 16Mb fiber optics Token Ring. The LANs and backbone will be connected by bridges/routers. Most networking hardware will be standard commercial products.

The UNK operator's consoles, 5-8 in total, will each be equipped with 3-5 graphics workstations and one server. The latter will provide common services (file, print, plot, etc.) and can also be used to execute certain run-time application software.

There will be a dedicated UNK control system's data base management server CDB and a general purpose server GPS for number crunching, modelling. The latter also caters for general user program development, thus supporting numerous workstations and terminals spread around the buildings.

3. SOFTWARE ARCHITECTURE

Experience in development of accelerator control system software packages world wide shows that certain functionalities are existing in one form or another in the majority of them. The general trend, which we try to follow, is therefore to extract these common parts and provide them as standard facilities which may be used by each *specific application* task. This would allow to eliminate multiple code and improve system reliability and maintainability. The software implementation of those common functionalities may be called the *application environment* (Figure 3).

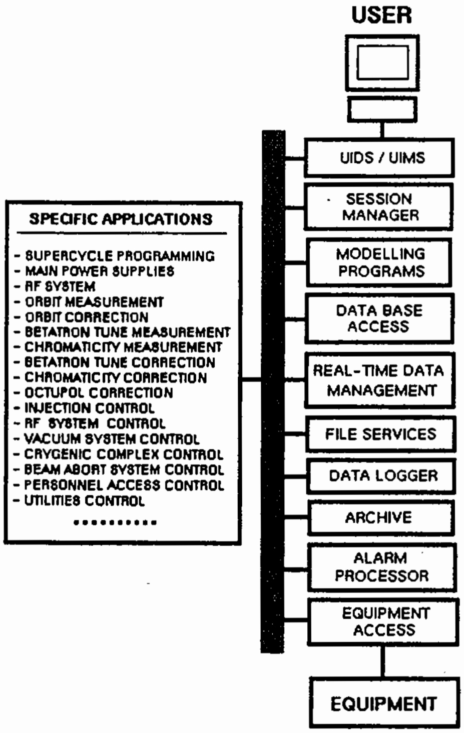


Figure 3 Specific Applications and Application Environment

In contrast with a general purpose application independent system software the application environment is problem (controls) oriented. It is supposed to be relatively stable and only evolve slowly during a control system's life cycle.

Application Software

A first analysis of the UNK control system functionality has been made using the SASD methodology. Thus the main procedures, data flows and data stores were identified. The application software functionality can be represented by the following broad groups of tasks.

1. Process modeling and preparation of data sets which determine the operation of accelerator devices. Advanced modeling programs can simulate a particle behavior for the different modes of operation of the accelerator and in such a way test a validity of the data set prepared.
2. Data down-loading, acquisition and trimming.
3. Routine surveillance of the technological accelerator subsystems (like electricity, vacuum, etc.) Alarm signal generation and processing, fault recovery.
4. Data logging and archiving services which allow to keep working track and history of the accelerator operation.

This analysis, design and data structure development will be pursued using a modern integrated CASE tool package. The latter should also provide project management and documentation support, which are important when numerous programmers of different level must cooperate.

System Software

The application environment is built on the basis of elementary functions provided by the system software. The general UNIX operating environment has been chosen for the UNK control system.

The UNIX (Ultrix) will be used in the operator workstations and servers while a Unix-like real-time sytem will be chosen for FECs. For this part the LynxOS from the Lynx Real-Time Systems Inc. is in the process of evaluation.

The TCP/IP, RPC, NFS packages, now a standard features of most Unix and real-time systems, will be used in the general network.

For the 8086 based ECs, two operating systems are considered now: the MTOS-UX from Industrial Programming Inc. and Intel RMX compatible DOS-86 which is available on the USSR software market. A final choice will be made by the end of this year. The LSI-11 based ECs will use an operating system similar to the DEC RSX-11.

User Interface

The user interface is based on graphical workstations and windowing techniques. Commercially available UIDS/UIMS (User Interface Development/ Management Systems) built on the basis of the X-Windows and OSF/Motif standards are considered now for the user interface creation and management (XFaceMaker, TAE+, Teleuse, VUIT). Such systems allow easy interface design and quick prototyping.

Much recent work in this field is concerned with specific extension of commercial products to a standard set of control tools and screen layouts which would allow to unify human interaction procedures for a wide range of applications. Early specification, prototyping and demonstartion of such a tools would allow to start application software development and meet user's requirements.

DBMS and Real-Time Data Management

Off-line data preparation, including descriptions of machine and control system objects and relations, will be done using the Oracle DBMS and related tools.

One presently considers organizing all data inside of the control system in a specialized home-made real time DBMS. It should contain both static read-only data, derived from Oracle and dynamic data of the current machine state, some number of pre-defined UNK states for pulse-to-pulse modulation (PPM) and accelerator development, etc. This real time DBMS will support access to read/write data for fixed and off-line prepared data structures. It will serve a number of distributed data bases. Each DB is a standard file, containing data in the form of three-dimensional tables, some of which may be duplicated in the memory of specified computers..

Equipment Access

The diversity and multiplicity of process devices, requires some uniformisation, i.e. hiding the device specifics from the operational applications. For doing so, the device specifics shall be encapsulated in software envelopes having a standardized access protocol.

This concept leads naturally to the object-oriented approach to logical equipment representation. An equipment object can model a real, physical unit of the equipment or abstract entity, like a feedback loop or a data buffer.

The applications' vision of equipment is strictly limited to a relatively small number of logical device classes. Each logical device corresponds to a physical component or group of components of the accelerator equipment able to perform a complete task and considered as a single entity in context of the accelerator operation. A device object class can represent, for example, an ion pump with all its attribures (e.g., status, voltage, pressure, etc.) and services, which it provides for an application task (switch on, set voltage, read pressure, etc.). Each particular ion pump in the system will be an instance of the ion pump class.

A logical device is a complex object that encapsulates a collection of cooperating component objects. Each component object class represents a stable partial functionality inside of a logical device (input/output, local data buffering and processing, surveillance, timing, etc.). The set of the component object classes forms a toolkit for logical device construction. The set of component objects constituting a device, with their relationships and dependencies, formally represent logical device model.

A logical device undergoes the following main phases during its "life cycle" in the control system.

1. Device class creation: positioning of the class in device class hierarchy, definition of new logical device attributes and services using the component objects class library.
2. Implementation: integration of standard software modules, corresponding to the library classes, and, possibly, writing of a code reflecting a new device specific features. It should be noted that the same logical device can be implemented in several versions on different hardware platforms.
3. Instantiation: creation of a particular device object which is an instance of the device class. The object identifier, implementation version and equipment network address are specified at this phase.
4. Initialization: downloading of the device software into the specified platform, initial test and device setting in a predifined initial state.

Commercial Control Software Packages

Standardization trends result in the appearance of "generic" integrated commercial controls software packages which can be configured to realize a wide range of functionalities. Examples are V-System from VISTA, G2 from Gensym Corp., GENESIS from Iconics, etc.

For the purpose of thorough evaluation the V-System software will be used as a kernel of a relatively small prototype control system . The system will support commissioning of the U-70 to UNK-I beam transfer line, planned for the middle of 1992. The V-System will run in the VAX/VMS workstation connected via Ethernet to front-end computers (MS-DOS IBM/PC). Each of these front-ends will control a certain number of technological subsystems (including all types of power supplies, beam instrumentation, vacuum and utilities subsystems like electricity, ventilation etc.) by using equipment controllers connected via RS-232 interface. These PCs will be used as local consoles for equipment tests at the begining and thereafter they will work as a "data pumps" to supply data for the V-System data base.

4. CRYOGENICS ASPECTS

The cryogenics and related equipment is a substantial part of the UNK project and falls into three broad groupings.

1. The cryogenics complex that provides cooling of the superconducting magnets placed in the UNK ring tunnel.
2. The quench protection system.
3. The superconducting magnet main power supplies with their ramping and dc programs.

By their nature these systems have a close internal binding and require only a weak coupling with the main UNK control system. A fair degree of autonomy and stand alone capability is therefore foreseen, which is helpful in commissioning and later servicing.

According to the functionality and geographical distribution of the equipment, the cryogenics complex control is subdivided among four relatively independent subsystems.

1. The gaseous helium storage, compression and purification control. The corresponding equipment is located in three compressor station buildings placed around the UNK ring.
2. The helium liquefiers control. There are 6 helium liquefiers located in the central helium liquefier building.
3. The satellite refrigerators and magnet cooling control. There are 24 refrigerators located in 12 buildings regulary distributed around the UNK ring. Each refrigerator supports operation of one superconducting magnet string.
4. The Nitrogen Storage and distribution control.

Local access to the cryogenics equipment will be provided via local consoles in each cryogenics complex equipment building. The local consoles will be used in the equipment commissioning, autonomous tests and troubleshooting. The normal operation will be performed from the Cryogenics complex Control Room (CCR) located in the central helium liquefier building. The CCR will be equiped with the same sort of tools as the UNK MCR and provide 5-7 operator's workplaces.

Cryogenics complex controls will, like the main UNK control system but with some special flavours, use the Multibus-I based ECs with 16 bit processors connected by MIL-STD-1553 to the standard UNK FECs.

A specialized Multibus-I programmable controller module (SPC) has been developed and will be used at the lowest control level in the cryogenic system. The module consists of the general purpose 16-bit Multibus-I processor card and a functional card which is connected to the processor via a private bus. There are few types of the functional cards developed for interfacing to various kinds of transducers and actuators used in the cryogenics equipment. The SPC will autonomously realize closed loop control and relay logic algorithms according to the program stored in its PROM. A Multibus-I crate may house a number of SPCs working in parallel under supervision of the main EC processor. The total number of SPCs in the cryogenics complex control will be more then 2000.

In collaboration with CEA, Saclay, a first version of a FNAL-like quench protection system will be tested on an experimental sector of 8 protection units, each consisting of 12 dipoles and 2 quadrupoles. The front-end electronics and emergency heating power supplies are located in shielded cavities in the ring tunnel.

The main power supply controls will form a closed subsystem, loosely coupled to the main controls and interlocked with the quench protection system.

5. UTILITIES AND SERVICE NETWORK

The UNK complex includes a number of various utilities providing, as a whole, the safe and reliable environment for the main accelerator subsystems and personnel working in the accelerator area. These utilities are the general electricity and water distribution networks, tunnel ventilation and gas analysis system, radiation protection, fire safety and personnel access control to restricted access areas.

The utilities controls shall be supported by relatively slow, but very reliable and uninterruptable control system. A special highly reliable service network will interconnect a number of regional control nodes with central utilities control room. The service network will be linked to the main control network and access to the utilities can be provided also from the MCR consoles.

The market study, evaluation of commercially available components and prototyping have been started. A part of this work is design and implementation of utilities control system for the beam transfer line from U-70 to UNK-I (BTL). The system is based on the standard industrial programmable logic controllers (PLC). The PLCs perform low level control and monitoring functions and are linked via RS-232 lines to the IBM PC compatible console computer in the temporary beam treansfer line control room. The IBM PC collects data from the PLCs, performs data analysis and generates alarm messages and interlock signals in case of faults. It makes also periodic data logging and keeps a log of the system operation.

6. EXTERNAL BEAM LINES

The external beam lines will comprise a 12 km long neutrino channel, leading up to the neutrino experimental area, and three 6 km long hadron beamlines leading to their own experimental area each. The operational patterns of the beamline zones are, by their nature, different from the accelerators. The essential aspect is the more frequent and rapid changes, following the experiment's requirements.

The technology of the equipment used in these beam lines and experimental areas is similar to the one of the accelerators proper, including the use of superconductivity hence cryogenics. A strongly different aspect, however, form the target areas and splitting stations with their radiation problems and remote handling requirements. Some advanced devices such as polarimeters and crystal bending and focusing may be used.

One presently considers a controls architecture which is very close to the one described for the main UNK ring. There will be workstations with the modern windows and graphics oriented software, these will be interconnected with a TCP/IP based network, which in turn connects to the accelerator network, and at the frontend to VME based 32 bit microprocessors driving ECs over the MIL-STD-1553 fieldbus. In contrast with the accelerator system, the ECs of the external beam zones controls may still be CAMAC based.

Powerful number-crunching multiprocessor assemblies will be used for digital signal and image processing in the polarization hodoscopes, electron-gammas tagging system and TV- cameras based beam instrumentation.

Systems software will essentially be the same as for the accelerators. Applications will be strongly data driven and model oriented and an expert system is being contemplated for operator support.

Although routine operation may be done from a central MCR, a strong local access and stand alone component remains essential.

7. U-70 CONTROLS UPGRADE

Contrary to the situation for the accelerator and beam zones controls, for which options were open, the U-70 injector group has a strong historical bias since both booster and U-70 proper are largely computer controlled. Moreover, the upgrade must be done virtually without interruption of the U-70 experimental programme. Finally, the urgency of this upgrade practically dictates using products now readily available in the USSR and using a stepwise implementation, converting small slices during the short planned shutdowns.

The FECs for the U-70 upgrade will be the Multibus I based SM1810.30 computer with an Intel-like 16 bit 8086/8087 processor board, using an RMX compatible real time kernel. They will have appropriate RAM capacities, a LAN interface and a parallel branch highway CAMAC driver. Each of the about 12 FECs, spread over 4 buildings, maximum 4 km apart, drives up to 3 CAMAC crates catering for I/O. Servers for files and data base, 4 to 6 in total, will be enhanced configurations of the FECs, featuring larger memory, hard disk and a more complete generation of the operating system. Interaction will be using PC-AT computers under DOS. A commercial LAN product is still being sought in the USSR.

8. PRESENT STATUS

A conceptual design study of the upper part of the control system has been made and is accepted. Prototype partial integrations of the main control system and front end assemblies, as well as a quench protection test facility, are being prepared and should be available early 1992. An applications development environment with servers and workstations is being prepared. The external beam zones controls are in the conceptual design phase. The U-70 controls upgrade has been largely defined and frozen and implementation has started.

The Multibus-I based ECs have been largely defined, prototypes of most modules exist, industrial contracts are being negotiated. All ECs for the BTL are assembled and tested and installation in the equipment buildings is in progress. The V-System applications for the temporary BTL controls are largely prepared and corresponding communications and PC front-end software is under development.

9. REFERENCES

1. M.Mikheev, N.Trofimov, E.Scherbakov, S.Zelepukin, B.Kuiper, "The UNK Control System Conceptual Design Study", IHEP and CERN internal report, June 1990.
2. V.Alferov, S.Goloborodko, S.Logatchev, A.Skugarevsky, "The Control System for the UNK Cryogenic Complex", IHEP internal report, November 1990.
3. V.Sakharov et al., "Preliminary Considerations on the Top Level Part of the Control System for UNK External Beam Channels", IHEP internal report, November 1990.
4. S.Balakin, V.Brook, V.Voevodin, V.Tishin, "The Control System for the Booster Synchrotron of IHEP", 19th School fpr Control Systems in Science Research, Novosibirsk, 1985, p.11.
5. V.Voevodin, "Preliminary Considerations of the Real-Time Data Management for the UNK Control System", UNK Control Group internal report, June 1991.
6. M.Mikheev, "Proposal for Use of the V-System Software in the UNK Beam Transfer Line Control System", UNK Control Group internal report, April 1991.
7. N.Trofimov, "On Object-Oriented Approach to the Equipment Access", UNK Control Group internal report, September 1991.

MOSCOW UNIVERSITY RACE-TRACK MICROTRON CONTROL SYSTEM: IDEAS AND DEVELOPMENT.

Chepurnov A.S., Gribov I.V., Morozov S.Yu., Shumakov A.V.,Zinoviev S.V.
Institute of Nuclear Physics, Moscow State University
119899, Moscow ,USSR.

Abstract.

Moscow University race-track microtron (RTM) control system is a star-shape network of LSI-11 compatible microcomputers. Each of them is connected with RTM systems via CAMAC; optical fiber coupling is also used. Control system software is designed on Pascal-1, supplemented with real time modules and Macro. A unified real time technique and reenterable data acquisition drivers allow to simplify development of control drivers and algorithms. Among the latter three main types are used: DDC methods, those, based on optimization technique and algorithms, applying models of microtron's systems. Man-machine interface is based on concept of the "world of accelerator". It supports means to design, within hardware possibilities, various computer images of the RTM.

INTRODUCTION.

Moscow University race-track microtron - when it's construction will be finished - is to produce 175 MeV 100% duty factor electron beam with low transverse emittance (0.05 mm*mrad) and up to 0.01% energy monochromaticity [1,2]. To support means for easy programming of microtron's behavior, when being adjusted, and to meet requirements of experimental work, computer-based control system is to be developed. It's configuration is shown on fig. 1

HARDWARE.

Each micro computer of the control system is a 1-PCB LSI-11 compatible machine (EIS,FIS CPU; 1 mips; 56 Kb RAM). In control station it's connected with CAMAC via JCC-11 compatible crate-controller. Among CAMAC modules the following types are used: output and input registers (standard and specialized), FET multiplexers, 13,14,16 bit ADCs, step motor drivers. To prevent inadmissible interference, control system is isolated electrically from accelerator's equipment. For this purpose optically coupled measurement devices are used. Their terminal modules can be of three types: 19 bit TTL transmitter or receiver, 16 multiplexed a dozen bit ADCs or eight 12-bit DACs. Control stations (three of them in operation now) are connected via RS-232C interface in a star-shape network, formed by concentrator station. The latter is also tied with host-machine, used for software development and system loading. Man-machine interface and data-bases station is linked up with network like a control station. Concentrator machine and control stations have no any extra memory storage except CPU RAM. Man-machine and data bases station includes two microcomputers. One of them supports graphics, another handles data bases and communication protocols. This station, as well as host machine, is supplied with disc memory - including electronic one. Alphanumeric displays (VT-100) and some other peripherals are also attached. Among them - RS-232C interface, allowing to link up control system with external computer or network. Man-machine interface will also be supplied with four infinite-turning knobs to facilitate manipulation with one, two or three dimensional objects (control parameters value, terminal cursors etc).

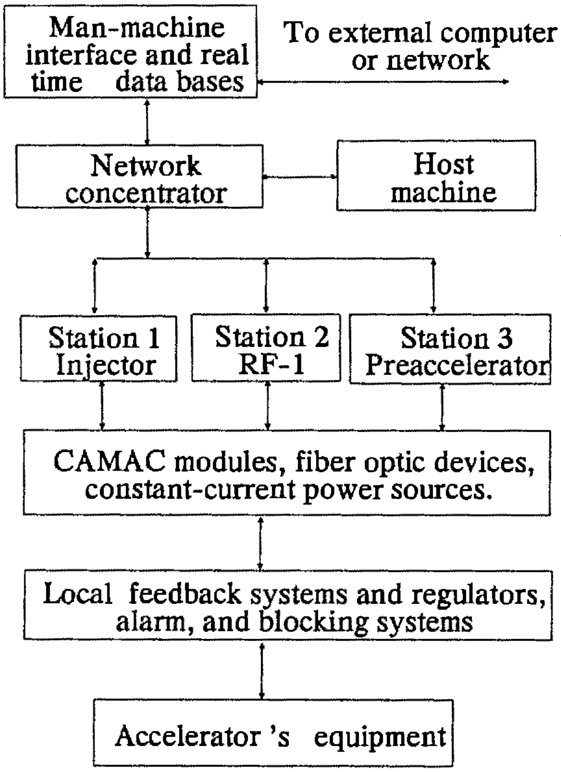


Fig. 1. Block diagram of Moscow University RTM control system.

SOFTWARE.

Compilation tools.

Basic compilation tools are shown on fig. 2. Source Pascal code with Macro insertions is being compiled with Pascal-1. Then a specially designed improver heightens an effectiveness of resulting Macro code. Afterwards, on Macro stage, it's being combined with CAMAC support modules. The resulting object code is being linked with Pascal library and the one, containing control system support modules.

Control station software structure.

The structure is depicted on fig. 3. Feedback control loops include control drivers, algorithms and reenterable data acquisition drivers. Reenterability allows to obtain measurement data by any program module, initiated with interrupt, while the same device is used by another program unit. Control drivers and algorithms are supervised with monitor. Real time processes deal with interrupts, initiated from different sources (network, CAMAC, timer). Network support system handles net protocol; access to data transfer buffer is also reenterable. Low level real time techniques include P-V operations to protect critical resources, repetition of critical section with non-savable resources and counting flags to prevent CPU overload.

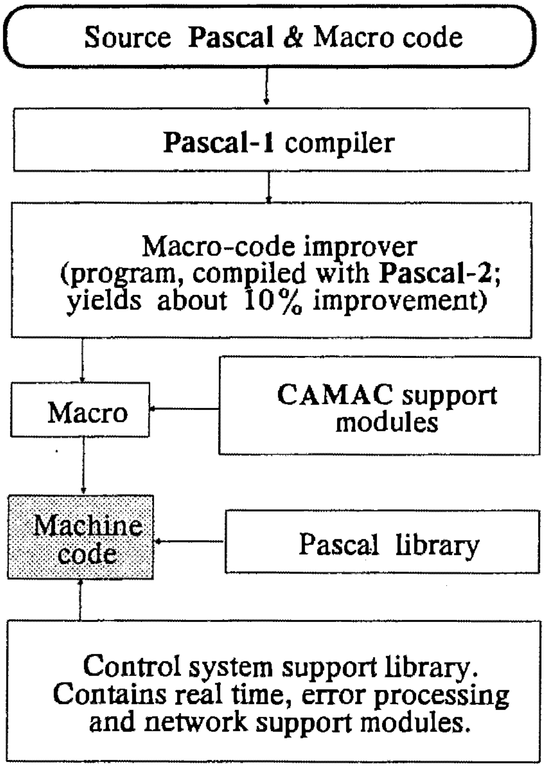


Fig. 2. Basic compilation tools.

Network data communication.

Data transparent network protocol is supported by exchange of byte-serial frames. They include command code, destination and source codes, data counter, unit of data and checksum. Command code indicates function to be accomplished with received parameters. These functions include: status of parameter inquiry, control, parameter value acquisition, free coefficients setting (calibration) and implementation of predefined operations. Special frames, consisted of only command code, are used to support network protocol and characterize general subsystem status.

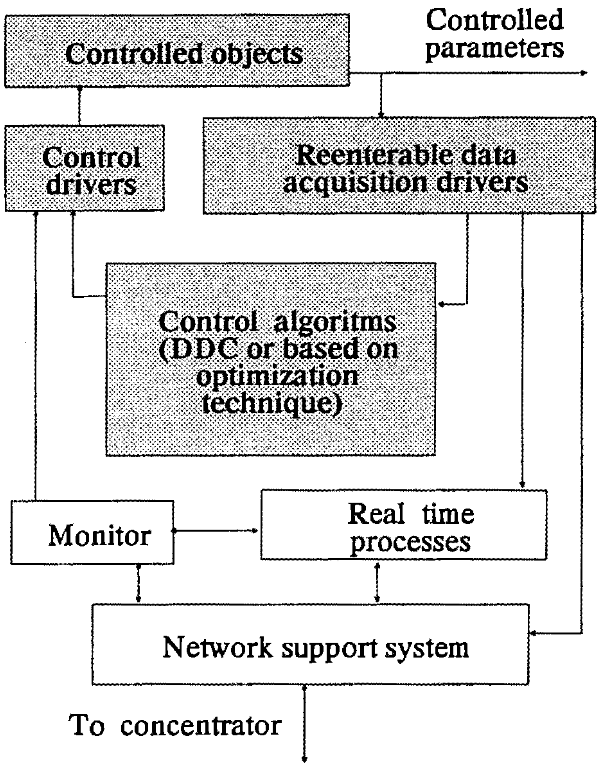


Fig. 3. Control station software structure.

Algorithms.

Algorithms, realized in RTM control system, can be divided into three groups: dynamic methods, functioning under strict time conditions, and two groups of quasistatic algorithms, where timing is not essential. The latter are based either on optimization technique or apply physical models of microtron's systems. Control system itself executes only dynamic and optimization methods. Dynamic ones include DDC methods both for logical control (switches) and analog algorithm simulation. The latter one (PI method) is used for temperature stabilization of accelerating sections [3]. Optimization algorithms are realized in one and two dimensional modes. One dimentional algorithm is based on "regula falsi" method and is used for fine tuning of various control parameters. in particular, to stabilize reference frequency generator. Two dimensional algorithm is used to steer electron beam via collimators by minimizing it's leakage current. The method deals with approximately symmetric goal function with flat bottom and uses non-derivative direct searching algorithm. At first stage steering algorithm scans with beam, using spiral trajectory, to find out goal function position. Model algorithms are not immediately supported by control system. They can be carried out with the help of any external computer, interfaced with control system. For this purpose man-machine interface computer also handles remote terminal protocol, allowing to inquire and set new values of any controlled parameter or to activate predefined functions.

Software development technique.

A considerable number of controlled parameters (about 400 now) requires definite technique to develop software. The development cycle is shown on fig. 4. It allows to avoid discrepancies between expression of requirements and actual control station program. During the cycle parameters data base is being corrected and supplemented. This base is used later to develop man-machine and "external world" program interfaces.

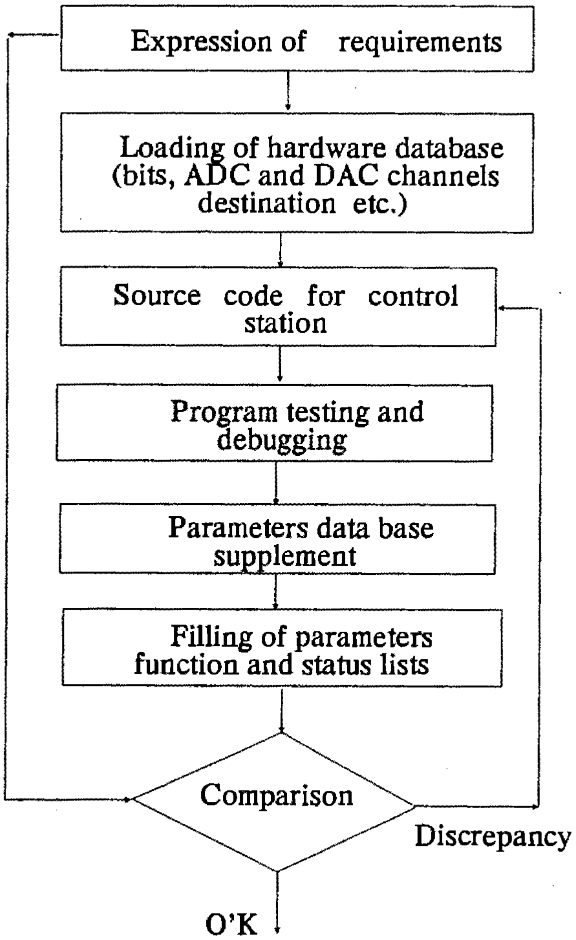


Fig. 4. Software of control station development cycle.

Man-machine interface and external world communication.

Man-machine interface software is based on approach, which gives means for operator to design - within hardware possibilities - his own computer images of the accelerator. It is supported by several types of windows and a list of their names (menu), allowing to activate any window. Special program supports creation of menu and windows. Graphical window can be chosen among several predefined types, which include those, representing several two-dimensional curves and the windows, which support an amplitude analyzers mode. Operator is not able to change geometrical shape of a window, but have a possibility to set colors, inscriptions and some other attributes. Any graphical window can then be loaded in a graphics support computer. Alphanumeric windows contain a set of parameters, extracted from parameters data base. Their values are represented on a terminal in any chosen position. Operator can previously fill the window with an arbitrary text. One type of alphanumeric windows is used to set constant parameters of accelerator (calibration coefficients etc). Their values are defined during a window's editing and transferred to subsystems when it's activated. Another type of windows - a working ones - support actual interaction with control system. They inquire and represent current values of parameters on a display. Operator is able to set a new value of any window parameter or initiate predefined control procedures. To adjust microtron's systems, it's possible to scan with any parameter (time among them), while others are represented graphically or (and) listed in a data-storage file. Working windows automatically support local data base files. Parameter's values are stored there before exit and extracted from the file when window is activated. There are also several windows, which are not programmable and are used to support system functions (time setting, password etc). Statuses of parameters, obtained from control stations, are stored in a special data base and can be displayed by operator's request. As mentioned above, man-machine computer also supports "external world" interface with a required data communication protocol.

References.

[1]. *A.S. Alimov, I.G. Artiukh, V.I. Shvedunov et al. Preprint NIIYAF MGU, 88-012/33, Moscow, 1988. (in Russian).*

[2]. *A.S. Alimov, V.G. Gevorkyan, I.V. Gribov et al. "Beam emittance forming line of the CW Race-Track microtron of the institute of Nuclear Physics of Moscow State University (INP MSU)". Nuclear Instruments and Methods in physics research A278 (1989)p. 379-388.*

[3]. *A.S. Chepurnov, I.V. Gribov, S.Yu. Morozov, A.V. Shumakov and S.V. Zinoviev "Feedback Systems for Local Control of Race-Track Microtron RF Accelerating Sections". These proceedings.*

Present Status of Control System at the SRRC

G. J. Jan*, J. Chen, C. J. Chen, C. S. Wang
Synchrotron Radiation Research Center, Hsinchu 30077, Taiwan
* also Department of Electrical Engineering, National Taiwan University, Taipei 10764, Taiwan

ABSTRACT

The modern control technique was used to design and set up a control system for the synchrotron radiation facilities at the synchrotron radiation research center (SRRC). This control system will be finally to operate the dedicated machine to provide the 1.3 GeV synchrotron radiation light. The control system will control and monitor the components of storage ring, beam transport and injector system. The concept of the philosophy is to design a unique, simple structure and object-oriented graphic display control system. The SRRC control system has the major features such as two level architecture, high speed local area network with high level protocol, high speed microprocessor based VME crate, object-oriented high performance control console and graphic display. The computer hardware system was set up and tested. The software in top level computers which include database server, network server, upload program, data access program, alarm checking and display, as well as graphics user interface (GUI) program were developed and tested. The operational system and device driver on the field level controller were implemented. The overall performance of the SRRC control system were tested and evaluation. The preliminary results showed that SRRC control system is simple, flexible, expandable and upgradable open system to control and monitor devices on the small scale synchrotron radiation facility.

I. INTRODUCTION

The 1.3 GeV synchrotron radiation facility is going to construct at SRRC to provide a low emittance and high brilliance light source. The dedicated synchrotron light source will include three subsystems that are a turn key 1.3 GeV electron full energy injector, beam transport line and storage ring. The 1.3 GeV full energy electron injector was constructed to Scanditronix AB at Sweden and installed at SRRC site. The injector is going to commission within 2 or 3 months. The beam transport line has been designed and its major component was also fabricated partially. The storage ring with triple bend achromat lattice [1] has been designed and the most components were constructed and passed its qualification of the specification.

The control system at SRRC provides a unique control and monitoring three subsystems which include the injector, the beam transport line and the storage ring facilities. The 1.3 GeV energy electron injector is composed of a 50 MeV linear accelerator and 1.3 Gev booster synchrotron accelerator. The control system of booster synchrotron can be run in a turn-key system and/or to be integrated into SRRC control desk to form an unique control system. The design concept is to standardize the same computer architecture, digital communication network with some protocols and some database structure as well as some console computer. The control system of the electron injector can play as a stand-alone subsystem to test and commissioning machine as well as machine study for booster synchrotron.

The control system of SRRC is cost-effectively designed by using recent developed computer technology [2,3] and modern control technique [4-9]. Two level hierarchical computers, process and console computer as a top level and multiple intelligent local controllers (ILC) as a low level, was configured. One process computer and several console computers at top level provide the database management and maintenance, devices control and monitoring, data logging and archiving, machine modeling, object-oriented graphical display. The lower level computers are multiple VME crate based system which handle device related data acquisition and control as well as local interlocking functions. It simplifies the architecture of control system at SRRC. The upper and lower level computers are connected by ethernet using high level protocol.

The process computer will handle the static and dynamic data base. It also offers to carry out the calculation of the electron orbit and simulation of the machine physics parameter. It is a high speed computing, multi-task and muti-user virtual memory system (VMS), and high through input/output (I/O) capability. The console computers will play a similar job as a process computer except maintenance of the database. The console computer plays an important role to operate and monitor the synchrotron radiation facility using man-machine interface that is developed based on the concept of the object-oriented graphic display. This is the most recent development on the third generation of the synchrotron radiation facility [7-9].

The ILC is a field level (or called device level) controller which performs the local data collection, local interlocking and closed loop control for the components and/or the equipment of the various system. It is also very important for the real time feedback control system.

II. COMPUTER HARDWARE SYSTEM

Two level hierarchical computer configuration has been designed and installed partially at this moment. The hardware configuration of the control system at SRRC is shown in figure 1. The top level computers are composed by one process computer and several console computers. The VAX 6000 model 610 supermini computer is chosen as a process computer. The VAXstation 3100 model 76 is selected as a console computer. The big semiconductor and mass storage capacity as well as high speed I/O peripheral devices are also considered and configurated.

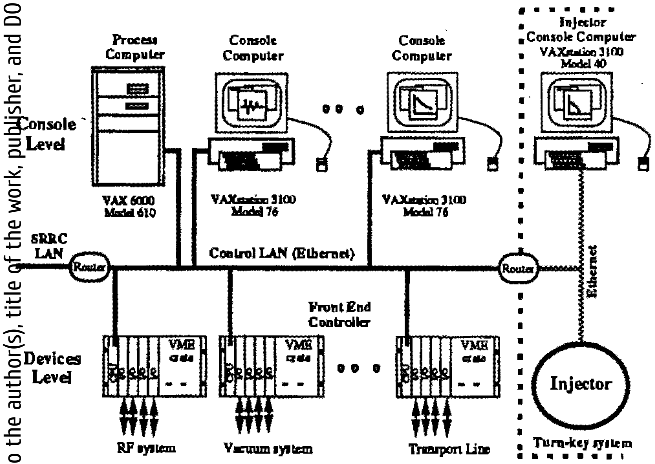


Figure 1. Hardware configuration of the control system at SRRC

The process computer creates and maintains the database and manage them at current development. The central database has been used at SRRC control system due to the on-time schedule to commission the machine. It will be modified as a distributed data base in near future. The update data rate from ILCs is set 10 Hz due to broadcasting mode or under request mode. These data can be received by any top level computers. The top computers can also set the parameters of the device and read the signal back and display it graphically. The process computer is running in VMS operating system but the console computer can run in VMS or Ultrix operating system. The ethernet using TCP/IP protocol is used to link all top computers and multi-ILCs.

The console computers are composed of VAXstation 3100 model 76, model 40 and DECstation 5000 model 200 that is run Ultrix operating system. The workstation will provide the console control and monitor the machine parameter and/or device signal through user graphic interface software. It provides the real time data trend or graphic display friendly.

The ILC is an VME crate system which includes Motorola MVME-147 central processing unit (CPU) board and variety of I/O cards. The CPU board consists of 68030 microprocessor , 68882 floating point processor, 4 Mbyte on-board memory and ethernet interface. The field devices are connected into ILCs through parallel analog or digital input/output or IEEE-488 bus standard or serial communication interface. The major tasks of the ILCs will handle the data setting, data acquisition interlocking, and close-loop control and monitoring function of the equipments or devices. Twelve set ILCs will handle the magnet power supply, RF system, vacuum system, beam diagnostic, and general purpose measurement instruments, ... etc. The user from the workstation can set and read back the parameters of the device easily.

III. COMPUTER SOFTWARE SYSTEM

Based on the VMS operational system and utility package, the software of the control system at SRRC has been designed and basic program of this software was coded and tested. The block diagram of software structure is shown in figure 2. The software structure is divided into several logical layers that is device access, network access, database management, graphics user interface and applications. The goal to modularize the software into layers is to reduce the developing time.

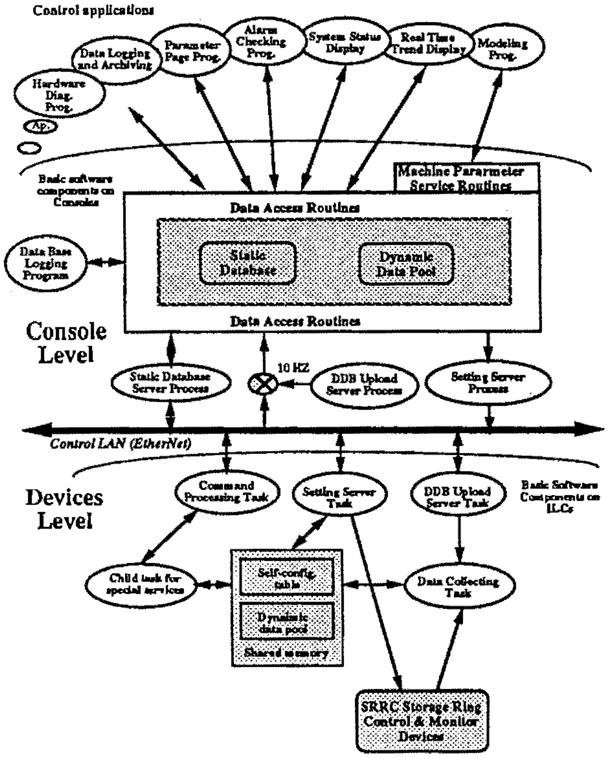


Figure 2. Block diagram of software system

The device access process are run on the ILCs. The pSOS+ real time kernel provides the ILC with support for task scheduling, memory allocation, even handling and message queuing. The pNA+ network support package provides socket network interface. The control tasks and various I/O tasks are developed and tested on the ILC. The device dependent software drivers are implemented partially. The magnet current regulator, RF low-level electronic controller and vacuum gauge controller were implemented and tested successfully. The speed of the dynamic data uploading to the top level computers is about 10 Hz. Downloading the database from the process computer into the ILCs to form a local distributed is initialized and underway.

The network access software is in charge of the information exchange between the top and the low-level computers. The protocol of the IEEE-802.3 is used to communicate with the turn-key injector system. The high level protocol TCP/IP is using at this moment. The reason to use IEEE-802.3 and TCP/IP protocols due to the beginning of the vendor contract to made an agreement for IEEE-802.3. That time the TCP/IP is not popular enough for VAX computer and 68000 series computer vendors. However, it would be changed into TCP/IP after the commission of the electron injector to form a unique local area network.

The database access is developed on the console level computers. The console level computers are VAX/VMS system (or can be an Ultrix system for the console computer). The software is coded in C language. The function of the process computer and operational console computer (or called workstation) is slightly different. The process computer keeps the system-wide static database and maintains it. The upload sequence is also requested by the process computer. At system start-up, each workstation requests and receives a copy of the static database from process computer. Each console computer then has all the database information necessary to process dynamic database frames received from the ILCs.

The workstation are mainly to provide user to operate the machine. The upload sequence is requested by the process computer, the ILCs multi-cast the dynamic data sequentially. All of the console level computer received dynamic data and updated into database at the same time.

The central database on the console level computer is used as buffer between the low level tasks at the ILCs and the console level applications. The application programs of the user can access the equipment parameters directly from their own database rather than from the ILCs. The application programs are device transparent. The development programs on the top level computers can be run concurrently.

The basic control and monitoring program has been developed at current stage. These programs are urgently need for machine commissioning. Those programs include the data logging, archiving, alarm checking and display routine etc., were coded and tested successfully. The machine model calculation can be run on process computer and/or individual workstation. The graphics user interface software was developed based on X Windows and Motif in VAXstation 3100 model 76. The block diagram of the relationship to develop the graphics user interface is shown in figure 3. The graphic edit program is to edit the display pattern of the machine components and build up the linkage relationship between the component and the static database. Those pattern file is stored with ASCII format in the hard disk. The control program is to read these ASCII file and make a connection between the component pattern and dynamic database. This program is also to execute the task of the data reading, setting and display from the DDB server. All subprogram is written in modularize software package. This human interface software will provide the operation of the synchrotron radiation facility more friendly.

IV. SUMMARY

The control system for the dedicated synchrotron radiation light at SRRC has been designed and the computer hardware and software was implemented partially since last year. Two level hierarchical computer control system has been configurated and the high speed local area network that use ethernet and high level protocol such as TCP/IP have been implemented and linked. The data upload rate is to maintain about 10 Hz without increase of traffic load at network. The over-all performance of the process computer and multiple ILCs as well as digital communication network has been tested and evaluated. The results show the 10 Hz update from the ILCs is not big issue to control and monitor the devices for synchrotron radiation facility at SRRC.

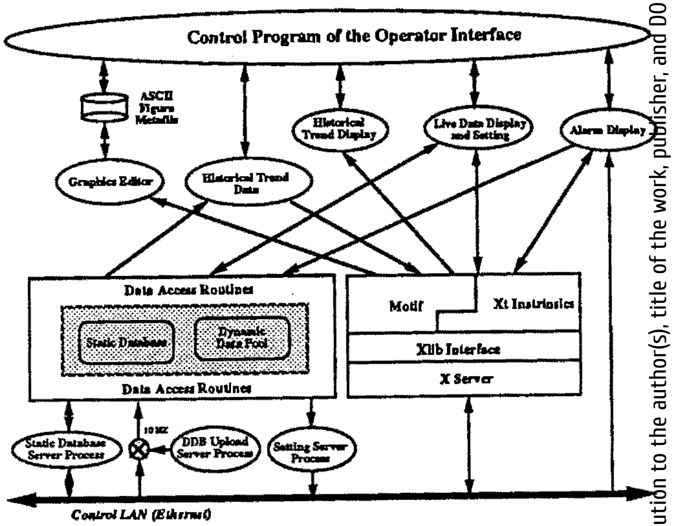


Figure 3. Block diagram of the operator interface

The object-oriented development tool of the graphic display and data trend display were developed and tested. The user can set and read back the signal from the field level devices within 10 Hz. The flexible change of the time interval for the data display at specified signal is allowable to change from 1 sec up to 60 sec. The GUI software can be developed from the developed basic edit program to code and modify it. This software will provide the user to operate the machine were easily and friendly. Finally, the two level computer architecture, multiple ILCs that are linked by ethernet local area network using TCP/IP, object-oriented graphic display software will be a nice control system for synchrotron radiation facility.

V. ACKNOWLEDGEMENTS

One of the authors (G. J. Jan) would like to express his appreciation to R. W. Goodwin and Mike Shea of FNAL, W. D. Klotz at ESRF and Tomotaro Katsura at KEK-PF for many useful suggestions. We also would like to acknowledge David Rice at CESR to discuss the technical consideration from the operational point of view. We would like to thank Shin-ichi Kurokawa at KEK-Tristan to give a lot of technical information and experience.

VI. REFERENCES

[1] "SRRC design Handbook", Synchrotron Radiation Research Center, April 1991.
[2] M. Castellano, S. De Simone, G. Di Pirro, S. Guiducci, P. Patteri, M. Serio L. Trasatti, "Diagnostic and control system for the LISA project", INFN, Laboratori Nazionali Di Frascati, European Particle Accelerator Conference, Rome, 1196 (1988).

[3] A. J. Kozubal, D. M. Kerstiens, J. O. Hill and L. R. Dalesio, "Run-time environment and application tools for the ground test accelerator control system", Nucl. Instrum. and Meth. A293, 288 (1990).
[4] S. Magyary, M. Chin, C. Cork, M. Fahmie, H. Lancaster, P. Molinari, A. Ritchie, A. Robb, C. Timossi and J. Young, "The advanced light source control system", Nucl. Instrum. and Meth. A293, 36 (1990).
[5] B. Robinson, and P. Clout, "Modularization and standardization of accelerator control systems", Nucl. Instrum. and Meth. A293, 316 (1990).
[6] J. P. Stott and D. E. Eisert, "The new ALADDIN control system", Nucl. Instrum. and Meth. A293, 107 (1990).
[7] W. D. Klotz, C. Hervé, "The conceptual design of the ESRF control system", European Particle Accelerator Conference, Rome, 1196 (1988).
[8] Marco P. Mignacco, "The ELETTRA control system", Nucl. Instrum. and Meth. A293, 44 (1990).
[9] F. Potepan, "The ELETTRA man machine interface", Internal Report ST/M-91/2, Sincrotrone Trieste, Italy, March 1991.

Status Report on Control System Development for PLS*

S.C. Won, S.S. Chang, J. Huang, J. W. Lee, J. Lee, J. H. Kim

Pohang Accelerator Laboratory,
POSTECH, P.O.Box 125,
Pohang 790-600, Korea

Abstract

Emphasizing reliability and flexibility, hierarchical architecture with distributed computers have been designed into the Pohang Light Source (PLS) computer control system. The PLS control system has four layers of computer systems connected via multiple data communication networks. This paper presents an overview of the PLS control system.

Introduction

The accelerator control system provides means for accessing all machine components so that the whole system could be monitored and controlled remotely. These tasks include setting magnet currents, collecting status data from the vacuum subsystem, taking orbit data with beam position monitors, feedback control of electron beam orbit, regulating the safety interlock monitors, and so forth. To design a control system which can perform these functions satisfactorily, certain basic design requirements must be fulfilled. Among these are reliability, capability, expansibility, cost control, and ease of operation.

Considering above requirements, the PLS accelerator topology, available resources, special accelerator hardware requirements and personal preference, we propose a hierarchical system architecture. To implementation of the control system, highly commercial approach should be made because of the tight construction schedule. Using well proven technology will promote reliability, and reduce the development effort. A development environment is set up to develop the prototype of beam close orbit correction system which is to damp up to 15Hz movements. The Beam close orbit correction system will use DSP(Digital Signal Processor) board for the fast computation. All BPM electronic modules and corrector power supply interface I/O modules for one acromat will be put into one VXI crate.

Hardware Hierarchy

To monitor and control the thousands of signals for PLS, it is desirable and cost-effective to establish a distributed control system based upon microprocessors. The PLS control system has a hierarchical structure as shown in Fig.1. The hierarchy consists of four layers, each of which has a different role. The four layers

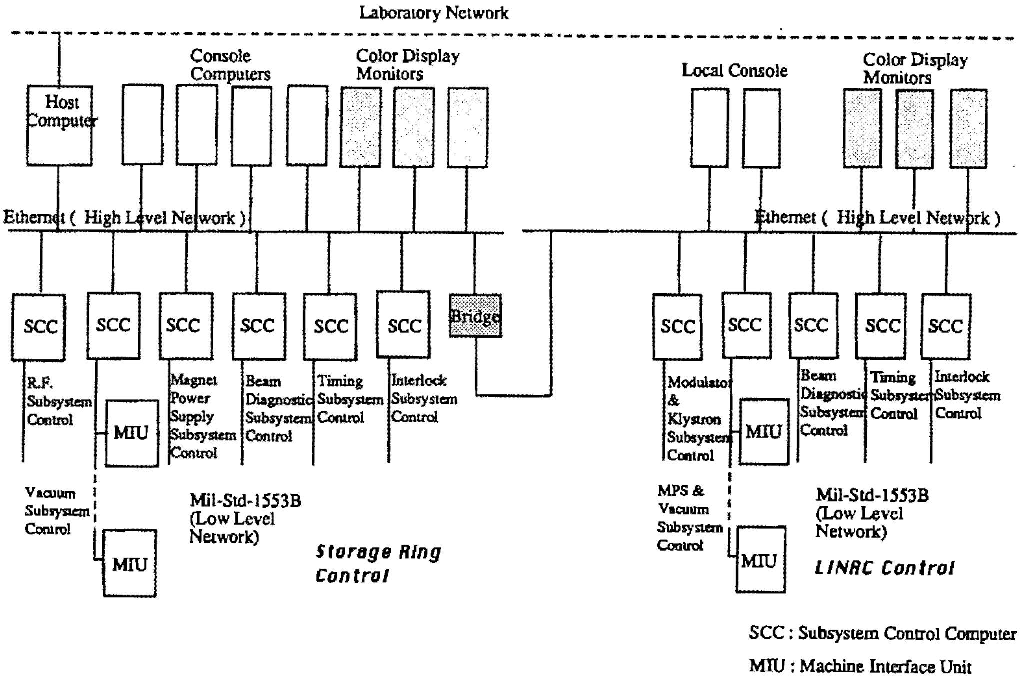


Figure 1. PLS Control System Architecture

---

* Work supported by Pohang Iron & Steel Co., Ltd. (POSCO) and Ministry of Science and Technology (MOST), Government of Republic of Korea.

are the computing service layer, the human interface layer, the subsystem control layer, and the machine interface layer. The computing service layer is a time-shared host computer in the computer room, and it is linked to various operator consoles which form the human interface layer. These workstations are located in the main control room. The consoles are connected to the master crates which form the subsystem control layer via an Ethernet. These master crates are located in local control stations distributed throughout the machine. The master crates are connected to slave crates, which form the machine interface layer, via a MIL-STD-1553B network. Each slave crate can use a variety of protocols to interface with machine components. Among these are IEEE-488, RS-232, RS-422. New protocols can be added with plug-in modules.

Computing service layer - Host Computer

A host computer with a high computational speed, large memory, and multiple job capability is needed for code and data management, mathematical modeling and simulation, off-line analysis, and for general purpose computing. For a low emittance light source, there are many critical parameters and it is more important to provide good computer modeling or simulation of the beam optics. For this purpose, the computation speed should be fairly high, and the machine should have at least a 32 bit processor with substantial physical and virtual memory space.

The DEC/VAX SYSTEM 6000™ may be a good choice for this as the VMS™ operating system provides an excellent software foundation upon which to run software donated by cooperative accelerator laboratories. However, if we can port the required software to the UNIX™ environment with little effort, a high performance RISC computer might be an alternative.

The host computer will be installed at the end of 1993 after the storage ring building is completed.

Human interface layer - Console Computers

Mid-range engineering workstations will be used as operator consoles to put in the operational parameters and show the operating status. Input parameters will go through some arithmetical processes and be converted into control commands for each affected piece of equipment.

Operating status should be shown on color display monitors in the form of simulation diagrams, various kinds of charts, and other graphical expressions. For this role, an engineering workstation is the best fit due to its high computing power and relatively low cost, high resolution color graphics capability, and high performance multiple window display system.

Currently, several SPARCstations™ of Sun Micro Computer Corp. are being used for console software development since they are well equipped with competitive capability, open system policy, and software availability.

Even though workstations provide excellent graphics capability, typical implementations of easy-to-use man-machine-interfaces have required sophisticated programming effort. However, we expect the use of a GUIDE (Graphical User Interface Development Environment) will be a great help in saving development effort.

DataView™ of V.I. Corp. is a potential solution. DataView™ consists of two components: DV-draw™, an interactive drawing editor for creating sophisticated screens without programming, and DV-tools™, a library of subroutines that connects graphics created with DV-draw™ to an application program. Thus it saves development effort, allows rapid prototyping, and future software interoperability derived from its support for industry-standard platforms. In addition, we will be able to easily integrate with other software environments such as relational databases and expert systems, and effectively develop integrated software by separating the GUI from application code.

A total of six workstations will be used as operator consoles for the storage ring and the linac, and a few additional color display monitors are planned for continuous display of information such as beam current, vacuum status, and magnet power supply current.

Subsystem control layer - Subsystem Control Computers

The Subsystem Control Computers(SCC) are microprocessor assemblies based on VMEbus and Motorola's 680x0 microprocessor. The reason for adopting VMEbus is its reliability and popularity. Each SCC consists of one Single Board Computer(SBC), an Ethernet interface module, and a MIL-STD-1553B network interface module, all of which are put together in a VMEbus crate. Motorola's MVME147™ with 32-bit 68030 CPU and 4 Mbyte memory is used for the SBC of the SCC.

The SCC does many control and monitor functions, such as reading and setting parameter values of machine components, feedback-control, alarm handling, and raw data processing for each subsystem, i.e., vacuum, magnet power supply, R.F., beam diagnostics, timing, and interlock. Each SCC is interconnected to the multiple Machine Interface Units(MIU) through MIL-STD-1553B network. The SCCs are also linked to the high level computers through Ethernet.

Machine interface layer - Machine Interface Units

MIUs are also microprocessor assemblies based on the VMEbus and Motorola 680x0 microprocessor family. Each MIU has one SBC, one MIL- STD-1553B network interface module, and a number of analog and digital input/output modules, all of which are put together in one or more VMEbus crates. An SBC equipped with Motorola's 68000 16-bit microprocessor and 1 Mbyte memory is used for the MIU. In order to handle various types of analog and digital input/output signals, a wide range of standard analog and digital input/output modules as well as home-made or non-standard interface modules are used. MIUs are distributed in local field stations, which are located around the machine, to reduce the length of the signal cables between the MIUs and the machine components and also to reduce electromagnetic noise problems with the cables. There are 12 local field stations around Storage Ring, 3 around Linac and 2 around BTL. Multiple VMEbus crates in the MIU are interconnected with VMEbus-to-VMEbus repeater module.

A development environment is being set up for the development of SCC and MIU. This environment is composed of development hosts and target VME assemblies, which are connected through Ethernet(TCP/IP). Motorola's SYS1147™ VME station, which uses MVME147™ as its single board computer, is used for the resident development host, and Sun Micro Computer Corp.'s SPARCstation/IPC™ is used for the cross development host. Microware's real-time operating system, professional OS-9™, is ported to the resident host and industrial OS-9™ is ported to every target VME system. Initially, this development environment will be used mainly for the development of the beam close orbit correction system and the modulator/klystron control system.

Data Communications Network

The components of the PLS control system are linked via two levels of data communication networks; a low level and a high level network. The low level network is used for data acquisition and forwarding of control commands. The high level network delivers operational setpoint values to the lower level computers and information acquired by the lower level computers to the console computers.

The low level network must be tolerant of electro-magnetic noise because MIUs should be installed close to various noise-generating equipment and cables. Due to this requirement, MIL-STD-1553B specification is used. MIL-STD-1553B is a multidrop network specification which is operated in a master/slave mode on which one Bus Controller(BC) may communicate with up to thirty Remote Terminals(RT). The SCC acts as the BC and the MIUs as RTs.

For the high level network, Ethernet is chosen because of its popularity which allows cheap and easy implementation for both hardware and software. TCP/IP will be adopted as the higher layer protocol for the same reason. Ethernet is a CSMA/CD network with a maximum transfer rate of 10 Mbps. However, the length of transmitted packet and transmission speed must be carefully selected to guarantee the appropriate transfer delay time and throughput.

Software

System software

A real-time operating system should be used for the SCCs and MIUs because some machine control jobs such as closed orbit correction requires real-time performance. OS-9^{TM} from Microware Systems is selected because of its advanced kernel functions and variety of software development tools. OS-9^{TM} not only provides a real-time kernel and its associated system modules, but all the file managers and device drivers necessary to support integrated i/o processing. The user interface includes an easy-to-use UNIX-like shell, hierarchical directory/file structure and over 70 utility programs to allow simple user access to the operating system's management functions.

The database on the SCC contains all subsystem parameters. It also has a component table which acts as a name sever providing the translation between the different symbolic names allowable at the high level and signal names in the MIUs, as well as linking these names with addresses of the appropriate MIU. The database on the MIU contains component data such as hardware addresses, set values, limits, conversion and calibration factors, and so forth.

Machine control software

Operating status of each subsystem such as vacuum, R.F., and magnet power supply subsystem should be displayed or archived to files, which might be later processed to analyze the operating history.

Desired setpoint values such as magnet power supply current and cavity gap voltage should be set. Setpoint values and other operational parameters might be put into the control system manually by the operator or automatically by the control software.

Intolerable differences between setpoint values and corresponding readback values should be continuously monitored and reported to the operator. These values are archived in files at the same time. The differences might be compensated by a slow feedback control program as long as there is no serious failure in correction. Continuous failure in correction would result in an alarm situation, which would be reported to the operator immediately. Any fault in apparatus should be reported to the operator, who can cope with it appropriately.

A total of 110 man-month is estimated to be necessary to develop the machine control software including the man-machine-interface, for both storage ring and linac .

Database

A comprehensive database defines all machine parameters and device signals. The database is generated in one of the console computers. The generated database consists of two parts: the static and the dynamic part.

The static database includes static machine parameters and device information such as names, locations, and various coefficients which might be used to convert scientific units into actual signal values, and vice versa.

Generated static database should be shared among the console computers by transferring the static part and maintaining consistency between the original and copies. Appropriate portions of the static database should be downloaded to the SCCs to be used to control each subsystem properly. The SCC might download parts of its static database to MIUs under its control. MIUs use this static database to convert scientific setpoint values into actual control signal values or actual monitoring signal values into scientific readback values.

The dynamic database consists of setpoint values such as magnet power supply currents and cavity gap voltages, and readback values such as ion gauge currents, magnet power supply currents and cavity gap voltages.

Dynamic database on a console computer has just a structure at the very beginning. It might be filled with valid data, when a control process which might need appropriate data were spawned. The valid data should be transferred from the appropriate SCC to the requesting console computer on "Supply-On-Demand" basis. Setpointing might also cause updating part of dynamic database with setpoint values.

For the PLS database structure, we took the idea of SPEAR's database system, since it is fairly portable and helped us to save development effort[4]. However, the update policy of dynamic data is quite different between the two database systems. PLS has distributed databases on the SCCs, the data of which can be supplied to upper layer computers on "Supply-On-Demand" basis. On the contrary, SPEAR has centralized database which is continuously updated by the hardware. We expect our scheme to maximize the network throughput by keeping unnecessary data from being transferred via network. Resulting gain in network capacity could be used for faster data acquisition of any particular signal group.

Beam diagnostic software

Knowledge of accurate machine parameters is very important for the efficient operation and study of the machine. We intend to automate all the beam diagnostic processes with various beam diagnostic programs such as beam orbit measurement, real time orbit correction, tune measurement, beam lifetime, beam emittance, and lattice function measurement, etc. Some diagnostic software will be run at the low level control computers for the real time beam diagnostics and feedback.

Modeling and simulation software

Traditionally commissioning of particle beamlines is a very time-consuming and laborious task. Even in day-to-day operation after start-up, various types of machine and beam errors have to be corrected. To reduce the time and effort for these tasks, fast and easy-to-use computer programs are needed.

Conclusion

An important requirement in designing PLS control system is flexibility. Natural growth of the system will require expansion of the control system or more complex control of the accelerator, and it is often necessary to implement new technology on the existing control system. We have designed a control system which makes substantial use of industry standard components, thus maintaining a very good level of flexibility and expansibility. we expect (and hope!) that industrial development of these standard component will procede faster than the ever increasing requirements on the PLS control system.

References

[1] S.C.Won, Jae W. Lee and Jinwon Lee, "Computer control system", PLS Conceptual Design Report, pp. 2-101, POSTECH, Korea, Jan. 1990

[2] S.C.Won, Jae W. Lee and Juno Kim, "Computer control system for PLS", Proc. of the 4th Asia Pacific Physics conference, Seoul, Korea, Aug. 1990, pp. 1114-1117

[3] S.C.Won, Jinwon Lee and Ji Hwa Kim, "DACS(Data Acquisition and Control System) for PLS", Proc. of the 4th Asia Pacific Physics conference, Seoul, Korea, Aug. 1990, pp. 1139-1142

[4] S. Howry, T. Gromme, A. King, M. Sullenberger, "A portable database driven control system for SPEAR", SLAC-PUB-3618, SLAC, Apr. 1985

Design of SPring-8 Control System

T. Wada, T. Kumahara, H. Yonehara, H. Yoshikawa, T. Masuda, and Wang Zhen
JAERI-RIKEN Spring-8 Project Team
Honkomagome 2-28-8, Bunkyo-ku, Tokyo, 113 Japan

Abstract

The control system of SPring-8 facility is designed. A distributed computer system is adopted with a three-hierarchy levels. All the computers are linked by computer networks. The network of upper level is a high-speed multi-media LAN such as FDDI which links sub-system control computers, and middle are Ethernet or MAP networks which link front end processors (FEP) such as VME system. The lowest is a field level bus which links VME and controlled devices. Workstations (WS) or X-terminals are useful for man-machine interfaces. For operating system (OS), UNIX is useful for upper level computers, and real-time OS's for FEP's. We will select hardwares and OS of which specifications are close to international standards. Since recently the cost of software has become higher than that of hardware, we introduce computer aided tools as many as possible for program developments.

I. INTRODUCTION

The SPring-8 facility consists of an 8 GeV storage ring of 1436 m circumference with a natural emittance of 7.0 nmrad, an injector linac of 1 GeV and an 8 GeV synchrotron. Figure 1 shows the layout of the facility. Construction starts in 1990, and the first stored beam is foreseen in 1998 [1]. The construction of control system will start in 1994.

From the designer's viewpoints, the control system consists of the following parts:

1) Computer System;
 1-1) Host and front end computers;
 1-2) Network system;
 1-3) Software:
2) Interlock System;
3) Analog Signal Observing System;
4) Timing System;
5) Television Network System;
6) Links with other System:

II. DESIGN CONCEPTS

Followings are the design concepts of SPring-8 control system:

1) Distributed processors system which are linked by high-speed networks.
2) Sub-system control computers are loosely coupled, due to different accelerator construction schedules and for the convenience of independent operation at maintenance time.
3) In normal operation, all the accelerators are operated at one control room by small number of operators.
4) VME and MAP for front end processors and networks, respectively.
5) PLC (Programmable Logic Controller) for fixed control sequence.
6) Real-time operating system for VME.
7) For high productivity of application program;
 7-1) Object-oriented programming;
 7-2) Computer aided program developing tools;
 7-3) Computer aided operation tools:
8) Standard hardware and software.

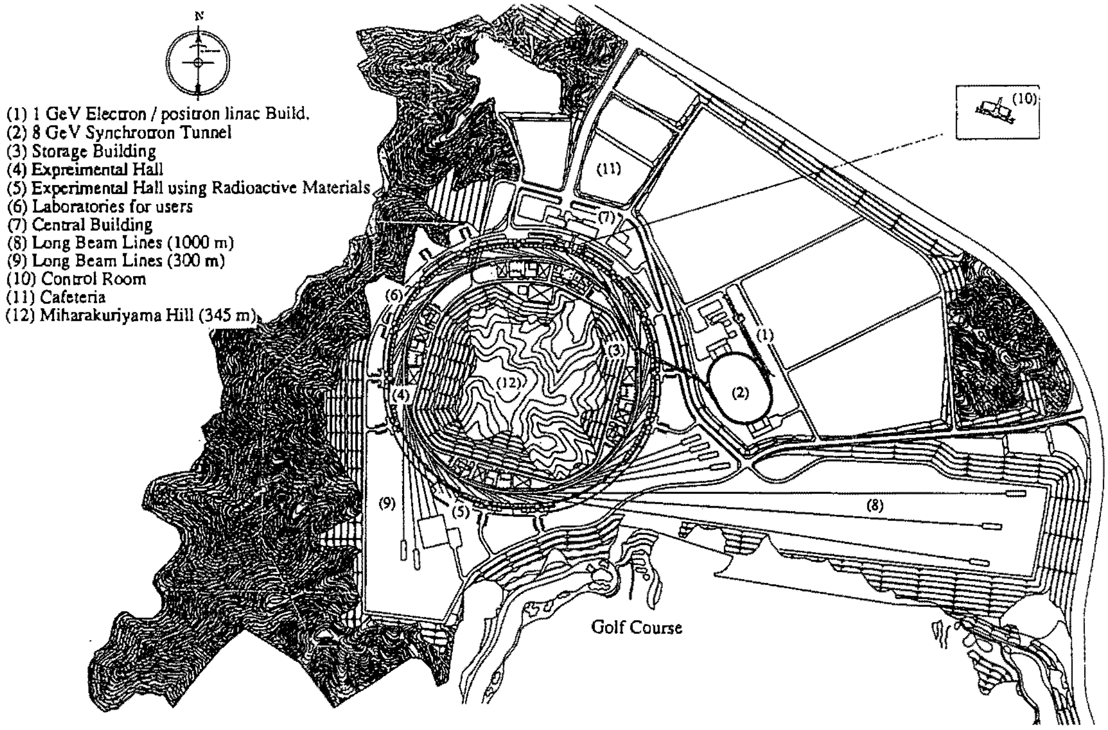


Fig. 1 Layout of the SPring-8.

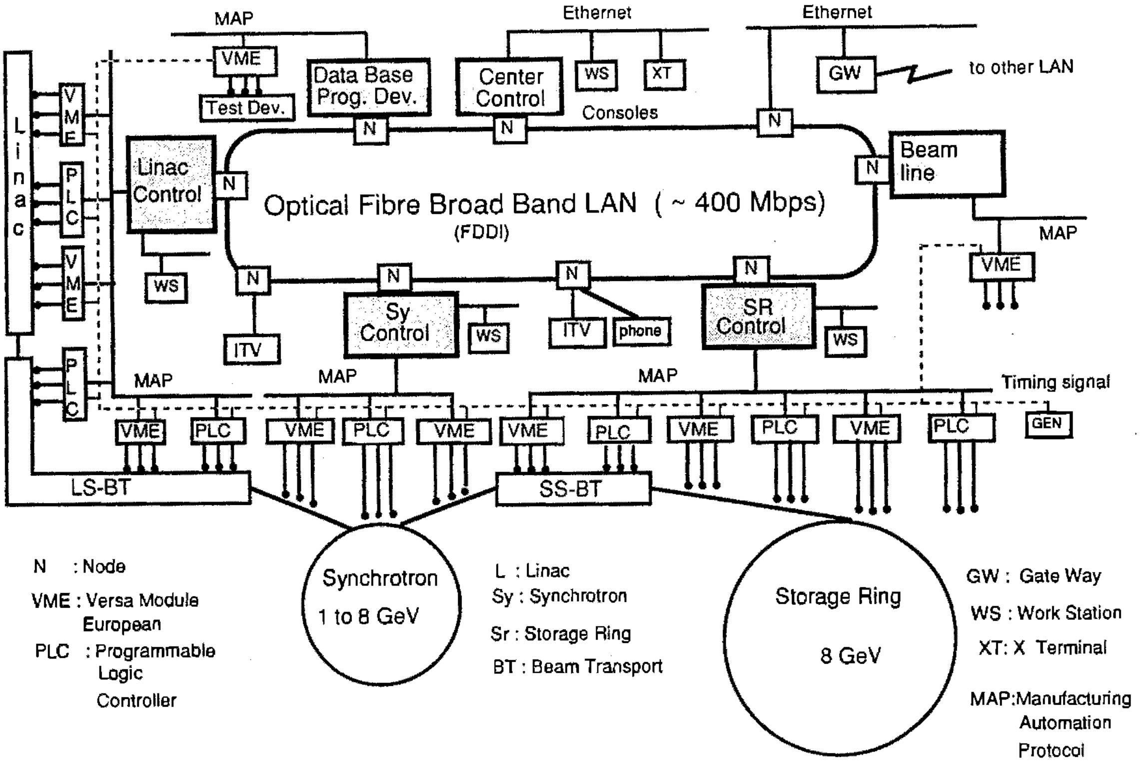


Fig. 2 The computers and their interconnecting networks.

III. COMPUTER SYSTEM

The control system consists of a central control system and several sub-system for each accelerator. These sub-system and central control system are linked through computer networks. Each sub-system consists of a host computer, several front end processors (FEP) and an operator's console. Figure 2 shows schematically the computers and their interconnecting networks. The function of each system are:

A. Central Control System

1) Selection of operating mode and scheduling;
2) Monitoring, logging, and display;
3) Linking with other computer system such as that of the computer center, of experimental system, and of a radiation safety control system.

B. Program development and Database

Central management of application programs and database:

C. Accelerator Control System

The control of injector linac, the synchrotron, and the storage ring. These sub-system controller are loosely coupled, due to the different accelerator construction schedules and the convenience of independent operation and maintenance.

At the lower hierarchy level, some FEP's are linked. These FEP's are microprocessors such as VME and programmable logic controllers (PLC). When high-speed data processing is required, such as by a fast digital feedback system, a VXI system will be used, since the data transmission rate of the VXI bus is much higher than that of VME system.

Workstation (WS) or X terminals are used as operator's consoles, which perform X-window servers.

The FEP's are distributed throughout the power supply rooms and on the circumference of the storage ring building, and the maximum length of the networks will be about 1 km. Table 1 shows the estimated number of input/output points, and the comparison with KAON facility [2]. Where LS-BT means beam transport line between linac and synchrotron, and SS-BT between synchrotron and storage ring. More than 200 VME system are to be used in whole control system.

Table 1 Estimated numbers of input/output points.

SPring-8

Part	Signals			
	DO	DI	AO	AI
Linac	1848	1128	672	96
LS-BT	62	92	0	37
Synchrotron	588	1057	86	203
SS-BT	725	397	63	142
Storage Ring	9113	15723	1382	3253
Total	12336	18397	2203	3731

TRIUMF

	DO	DI	AO	AI
KAON	7973	16310	2554	10214

IV. NETWORK SYSTEM

As is shown in Fig. 1, a so called multi-media LAN will be adopted for backbone control, with full- and mini-MAP LAN for real-time control. Voice signals and TV signals are also transmitted on FDDI (Fibre Distributed Data Interface) cables. MAP has two advantages over Ethernet (TCP/IP). One is that messages are exchanged by token-passing method. Hence, even under heavy traffic conditions, real-time response can be maintained. The other is that mini-MAP bypasses the middle 4 layers (3~6) of the 7 OSI layers. Thereby quick response is achieved.

V. SOFTWARE

A. Operating System

An UNIX is used for upper level computers, and real-time OS for FEP's. Table 2 shows examples of commercially available real-time OS[3]. Some of them conform to IEEE POSIX (portable operating system interface for computer environments) standard. An actual performance measurements have been reported on four kernels, all running on the same hardware platform [4].

B. Language

The candidates are Pascal, C++, and Ada which are appropriate for object-oriented programming. C and C++, however, are the main candidates because many program developing tools prepare C library functions as application programming interface. FORTRAN is useful for large numerical calculation programs, such as COD corrections.

For the effective development of application programs, it is convenient to use computer aided software development tools, such as CASE tool and GUI developer.

C. Artificial Intelligence

We intend to use expert systems mainly for alarm message handling system, and partly automatic operations. Introducing an AI tool, "NEXPERT OBJECT", a feasibility study of an expert system for the operation of a test stand of a klystron is started.

VI. TIMING SYSTEM

The common clock (508.58 MHz) is distributed with temperature compensated optical fibers to the linac, the synchrotron, and the storage ring. Trigger request pulses are to sent from the storage ring to the synchrotron and from synchrotron to the linac. Received devices generate beams after accurate delay time. We are developing an accurate delay pulse generator using GaAs preset counter circuit.[5].

VII. R&D

1). Development of accurate timing system for a 508 MHz clock.
    GaAs preset counter,
    E/O and O/E for optical fiber cable with small temperature coefficient.
2). Development of a pattern generator for the synchrotron operation.
3). Development of a field level bus.
4). Control of a klystron test stand.
    HW: EWS, X-terminal, VME, Ethernet,
    SW: UNIX, LynxOS, C, TCP/IP (socket),
    GUI: Xt, ("SL-GMS"),
    AI: "NEXPERT OBJECT"

VIII. REFERENCES

[1] H. Kamitsubo, "8 GeV synchrotron radiation facility project in JAERI-RIKEN SPring-8 Project", Nucl. Instr. and Meth. vol. A303, pp. 421-434, 1991.

[2] W, K. Dawson, R. W. Dobinson, D. P. Gurd and Ch. Serre, "A Conceptual Design for the TRIUMF KAON Factory Control System", TRI-87-1, 1987.

[3] Partly from T. Imai, Nikkei Electronics, no. 517, pp.149-153, 1991 (in Japanese).

[4] K. Low, S. Acharya, M. Allen, E. Faught, D. Haenni and C. Kalbfleisch, "Overview of Real-Time Kernels at the Superconducting Super Collider Laboratory", in 1991 IEEE Particle Accelerator Conference, San Francisco, CA, May 6-9, 1991.

[5] T. Kumahara, T. Wada, H. Yonehara, and H. Yoshikawa, "Design Concepts of The SPring-8 Control System", in 7th Conf. on Real Time Computer Applications in Nuclear, Particle and Plasma Physics, Julich, June 25-28, 1991.

Table 2 Examples of real-time operating system.

OS Name		VRTX32	pSOS+	VxWorks	VMEexec	C EXECUTIV	OS-9000	LynxOS	PDOS
Company		Ready Systems	Software Components	Wind River	Motorola	JMI Softwa Consultant	Microware Systems	Lynx Real-Time S	Eyring Research I
Microprocessor		680X0,80X86 SPARC	680X0 80386	680X0	680X0 88000	680X0 80286	80386,1486	80386,680X0 1860,88000	680X0
Bus		not fixed	not fixed	VMEbus	VMEbus	not fixed	not fixed	not fixed	
Network		Ethernet	Ethernet	Ethernet	Ethernet	Ethernet	Ethernet	Ethernet	Ethernet
Protocol		TCP/IP	TCP/IP	TCP/IP	TCP/IP UDP/IP	TCP/IP	TCP/IP	TCP/IP	TCP/IP
Kernel Size(KB)		8	11.5~ 13	60~ 465	8.5 min.	5~ 10	53	190	25
Task	Max. no.	256	65535	65535	No Limit	32767	No Limit	No Limit	
	Priority	256	255	255	256	32767	65536	32	225
Multi-Processor	Bus	○	○	○	○	×	×	○	○
	Network	○	○	○	○	×	×	○	
File System		MS-DOS	Dedicated, MS-DOS	RT-11,UNIX, MS-DOS	Fast File System	MS-DOS	Dedicated, MS-DOS	1-node	
Exec. Time	Switching	15μ s	19μ s	17μ s	19μ s	17μ s	220μ s	13μ s	26.2μ s
	Interrupt Wait Time	10μ s	7μ s	8μ s	6μ s	NA	40μ s	30μ s	
	Measured Condition	68020,25MHz 1 Wait	68020,25MHz no Wait	68020,25MHz no Wait	68020,25MHz no Wait	68020,25MH	80386,20MHz	80386,33MHz	68020,20MHz
AP Language		C,ASM,Ada	C,ASM,Ada	C,ASM,Ada FORTRAN	C,ASM FORTRAN	C,ASM	C,ASM	FORTRAN,C ASM,Ada	FORTRAN,C ASM,PASCAL
Environment for Devlopment		UNIX,VMS	UNIX	UNIX	UNIX	ANY	OS-9,UNIX OS-9000	Self UNIX	Self UNIX,PC

Design of a Control System of the Linac for SPring-8

H.Yoshikawa, Y.Itoh and T.Kumahara.
Accelerator Systems Development Group.
Office of Synchrotron Radiation Facility Project.
Japan Atomic Energy Research Institute.
Tokai-mura, Naka-gun, Ibaraki-ken, JAPAN.

Abstract

The design of a control system of the linac which is a large scale system including many unstable components like klystrons and modulators. The linac for SPring-8 requires to be operated automatically for injection to the synchrotron. Under these conditions, we chose a distributed control system architecture of a single layer net-work to simplify the protocol of the net-work between the linac, the booster synchrotron and the storage ring. A VME computer of 68030 is put in every modulator of the linac, and all control signals are gathered to the nearest VME computer. OS-9 and OS-9000 are on trial for investigation of the performances. TCP/IP is tentatively chosen as a protocol of the net-work, but we expect that MAP/MMS makes a high performance, and we are preparing a test of it.

INTRODUCTION

We decided that all hardware should be selected from ready-made machines for security of reliability, and our needs is satisfied at low cost without any customizing modules. Because this linac will be used on commercial base, reliability is the most importance for this control system. Moreover, easiness to use is necessary as a user oriented system. This linac consists of 26 pairs of a accelerator section and a klystron, so control signals mainly exist around modulators. Every signal is connected to the nearest VME computer in the modulator. 26 VME computers are connected to the flat network of one layer. Each VME computer works for a modulator, magnet power sources, vacuum pumps, RF phase control and monitors. Software of 26 VME computers are almost same, and it is easy to check whole performance of this system by a test bench of one set.

CONFIGURATION OF THE SYSTEM

A.Hardware

MVME-147s(Motorola) is selected as a CPU board, and it is set in a cage of 20 plots. Digital input/output boards are photo-isolated type, and analog input/output boards are two type of 12bit and 16bit. All signals directly come to the computer through a interface circuit of no CPU, but signals of monitors must have interface devices to establish satisfactory performances.

B.Software

The operating system is OS-9, and the language is C, and partly assembler is used. To select OS, VxWorks, VRTX, LynxOS, OS-9000 and others are check up. OS-9 is selected at the points of reliability, ability of stand alone work without development systems and suitability for bottom up build of a system. Applications and libraries will be made from practical use of object oriented programming.

STRUCTURE OF PROCESS

Every task consists of a control process, file-managers and device drivers. To get higher flexibility, all parameters are described in several parameter-files, and processes have no inner parameters.

A.Communication process

The data format of communication between processes is the same as the format of data through the net-work. If one process send a event (or a signal) to the out of one's cage through a network, the event is received by a communication process. The communication process searches the network address of the cage to which the target process belongs, and it send the event signal as a datagram to the cage. The address of machines are not fixed. The machine address resolution procedure(MAR procedure) defined machine addresses. When a searched process name belongs to a machine, the MAR process of the machine answers its address by broadcast.

B.Logging process

Most of the data sets are logged by a double buffered method. A double buffered method is combination of a ring buffer and a event buffer. The ring buffer stores continuous data of short interval, and if any troubles appear, shifting of the ring buffer is stopped to trace of the origin of the troubles. The event buffer stores log of status for long term.

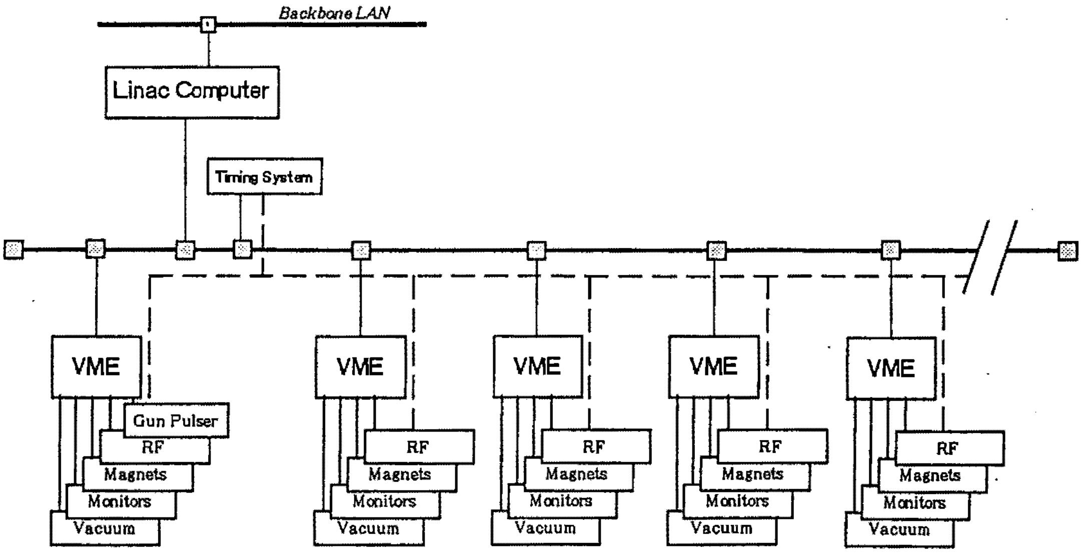
C.Interlock process

Inter lock signals must be taken by hard wires without computers. But it is able to reduce the wires by a inter lock bus system. In this linac, 26 modulators are almost same, and inter lock signals are classified into different emergency levels. The number of hard wires is the number of the emergency levels, and each hard wire is connected in series with same emergency level signals. Inter lock process on the computer sends the detail of what kind of inter lock works to the host computer.

CONCLUSION

Conceptual design of the linac control system is almost finished, and development of the system for a test stand will be started from next June. Coding of software is started already, and now modification of network protocol is going on.

Linac Control System Diagram



CONTROL SYSTEM FOR HIMAC SYNCHROTRON

T. Kohno, K. Sato, E. Takada, K. Noda, A. Itano,
M. Kanazawa, M. Sudou, K. Asami[*] R. Azumaishi[*] Y. Morii[†]
N. Tsuzuki[‡] H. Narusaka[§] and Y. Hirao

National Institute of Radiological Sciences
4-9-1 Anagawa, Chiba-shi, Chiba 260, Japan

Abstract

A control system for HIMAC synchrotron has been designed. The system consists of a main computer, console workstations, a few small computers and VME-computers connected via Ethernet. The small computers are dedicated to the control of an injection line, an extraction line and an RF system. Power supplies in main rings are controlled by the VME-computers through FDI/FDO, DI/DO modules. This paper describes an overview of the synchrotron control system.

INTRODUCTION

HIMAC is a heavy-ion accelerator complex for the clinical treatment of tumors and now under construction at National Institute of Radiological Sciences. It consists of an injector, a synchrotron, a high-energy beam transport and an irradiation sub-systems. Heavy-ions with a charge-to-mass ratio as small as 1/7 are accelerated up to 6 MeV/u through an RFQ and Alvarez linacs and injected to the synchrotron sub-system.

The synchrotron sub-system has an injection line, an extraction line and a pair of separated function type synchrotron rings with almost the same structure. These rings operate independently at different energies and same ion-species except that power supplies of two rings are 180° out of phase each other. The output energy of each ring is designed to be variable in a range of 100 - 800 MeV/u for ions with $q/A = 1/2$. The general description about the HIMAC synchrotron was given in the previous article[1].

An overall control system for HIMAC consists of a supervisor computer and four sub-system control computers connected via Ethernet as shown in Fig.1. The supervisor computer is used for the global control of the whole system of HIMAC. It is also linked by hardware and/or software to the other equipments in this facility such as a water-cooling system, an air-conditioning system and a radiation safety system. The sub-system computers control individual devices and carry out programmed sequences for device groups. The control system for the injector was already reported[2]. The control system for the synchrotron is also designed in the same manner, that is, 1) each system is operated rather independently by reducing the data to be transferred each other, and 2) hardware and software concerned with a man-machine interface must be standardized among all sub-systems.

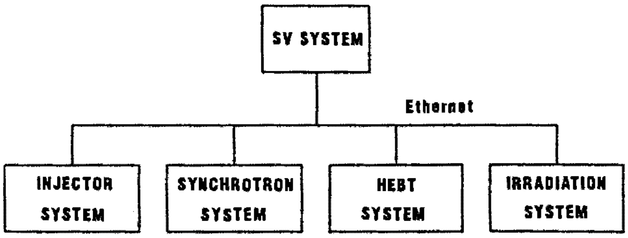


Fig. 1: A total control system for HIMAC consists of a supervisor computer and four sub-system control computers.

SYSTEM CONFIGURATION

The synchrotron sub-system has many devices of different operational characteristics, for example, dc power supplies in the injection and the extraction lines, high-voltage power supplies in the RF system and the power supplies in the main rings operated with patterns or pulses. In order to handle these devices of different types efficiently and to reduce the load of the main computer, we adopted a distributed and hierarchical structure. A schematic view of the synchrotron control system is shown in Fig.2. Components and their functions are as follows.

A main computer and console computers

A main computer serves mainly as a man-machine interface and a file server. DEC VAX4000/300 is proposed for this purpose under VMS with 64 MB memory, 3 GB disk and communication interfaces for RS-232C, GP-IB and Ethernet. In the HIMAC system, parameters such as current values, current patterns, timing relations etc. are saved as a parameter file in the main computer and referred as the reference data in the next operation of the identical condition. The main computer has to manage this database and carry out programmed start-up and shut-down sequences using these files.

For a man-machine interface two operator consoles are available corresponding to two rings. Two VAX Station

---

[*] Omika Works, Hitachi, Ltd., Hitachi-shi, ibaraki 319-12, Japan
[†] Toshiba Corp., Chiyoda-ku, Tokyo 100, Japan
[‡] Fuchu Works, Toshiba Corp., Fuchu-shi, Tokyo 183, Japan
[§] Digital Equipment Corporation Japan, Toshima-ku, Tokyo 170, Japan

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S03SRD14

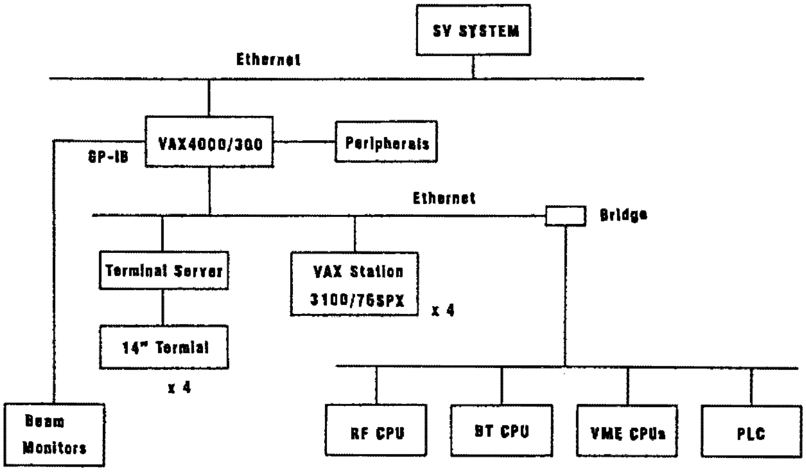


Fig. 2: A control system for HIMAC synchrotron.

3100/76SPXs with 16 MB memory and 104 MB disk and two 14-inch terminals are installed at each console for operation and display. Touch panels and rotary encoders are also equipped at the consoles and almost operations are performed by them with common procedures to the other sub-systems.

RF computer

A dedicated small computer connected via Ethernet controls the RF system including position monitors. Details of the RF control system are described elsewhere[3].

BT computer

The injection line from the injector sub-system and the extraction line to the high-energy beam transport subsystem are also controlled by small computers. Devices in both lines are mainly dc magnet power supplies, beam monitors and vacuum pumps except for a pulse magnet power supply to switch the injection beam to the upper and the lower rings. Device status, parameters and measured data are transferred through Ethernet.

PLC

A Programmable Logic Controller(PLC) is used as a interface between VAX4000/300 and the power supplies in the main rings to communicate ON/OFF commands, status signals and warning signals, although device parameters and measured data are transferred through DI/DO, FDI/FDO modules and VME-computers.

Power supply controller

All power supplies in the main rings are controlled by power supply controllers which consist of micro-computers and Digital I/O, Fast Digital I/O modules based on the VME standard. We plan to use 68000 family CPUs and a real-time, multi-tasking operating system, PDOS.

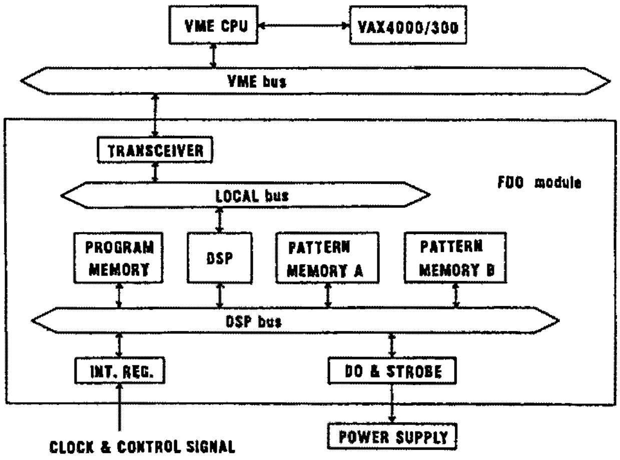


Fig. 3: A blockdiagram of the FDO module. The FDI module has almost the same structure.

The VME-computers communicate with VAX4000/300 via Ethernet and control dc power supplies through the DI and the DO modules.

On the other hand, the FDI and the FDO modules have been developed to control power supplies which should be operated with a pattern, such as a bending, a quadrupole and a bump magnet power supplies. A blockdiagram of the FDO module is shown in Fig.3. The FDO module has a Digital Signal Processor(DSP) and two pattern memories(A,B). The DSP sends data from the selected pattern memory to a power supply synchronizing with an external clock of 1200 Hz. While the DSP is reading and sending the data on the selected memory, new data can be written on the other memory, and then the memory is switched quickly from one to the other. The FDI module has almost the same structure and functions. This func-

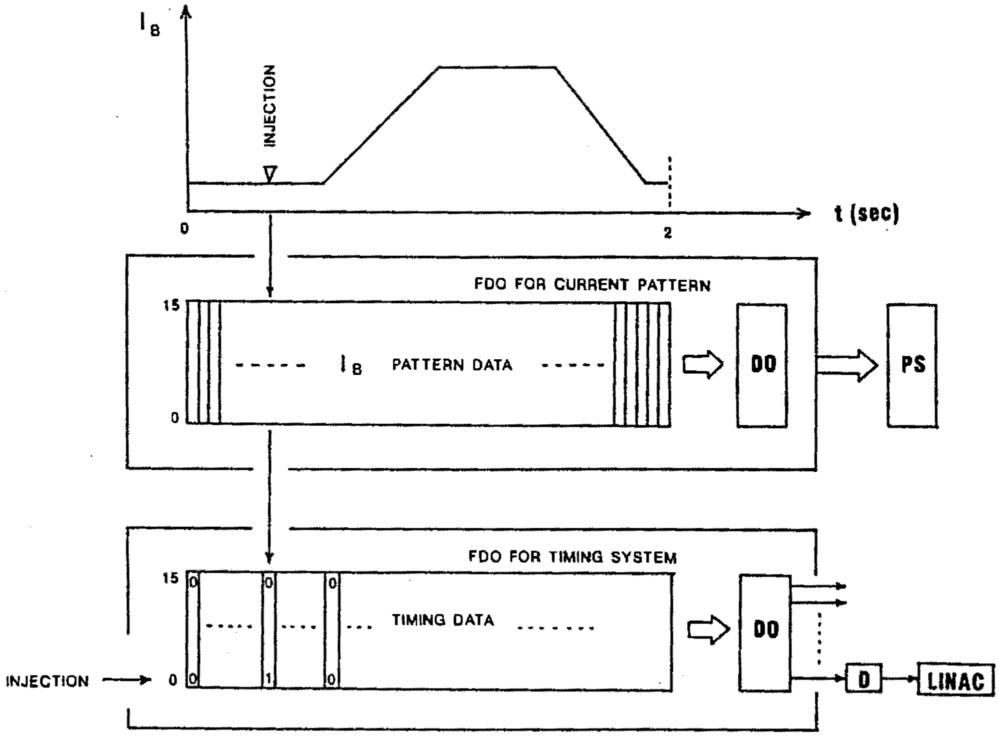


Fig. 4: A schematic view of the timing system. The FDO module is also used here.

tion plays an important role for a repetitive feed-forward pattern control of the bending and the quadrupole magnet power supplies in the main rings. In the feed-forward pattern control, the FDO module sends the reference data and the FDI module receives the measured data. Then the VME-computer calculates the correction pattern from the difference between the reference and the measured data using a transfer function of the magnet system and writes the new pattern on the background memory while the foreground memory is facing the power supply.

TIMING SYSTEM

A timing system is very important to synchronize the sub-systems because each sub-system is designed to be operated rather independently except for a software interlock among the sub-systems. Event signals are generated in the synchrotron sub-system and delivered to the own and the other sub-systems by hardware. As the bending magnet power supply consists of 24-pulse thyristor rectifiers and must be operated in synchronization with the ac power line of 50 Hz, the timing system generates clock signals of 1200 Hz (50 Hz × 24 pulses) phase-locked to the ac line voltage. The FDO module is also used for the setting of the event signals. A schematic view of the timing system is shown in Fig.4, where the synchrotron is operated, for example, with the repetition rate of 0.5 Hz. The FDO module for the current pattern data of the bending magnet has 2400 data of 16 bits and they are transferred in synchronization with the clock of 1200 Hz. The FDO module for the timing system has the data of the same size. In this case, however, not the 16 bit data but the rows of 0th - 15th bit have meaning because each bit is assigned to a specific event, for example, 'BEAM INJECTION', 'RF ON', 'ACCELERATION START', etc. The event signal corresponding to each bit is generated at the time when the bit is 'ON' or '1' and delivered through a delay controller.

ACKNOWLEDGEMENT

The authors wish to thank many of their colleagues concerned with the HIMAC project for their helpful discussion and advice.

REFERENCES

(1) K. Sato *et al.*, "Heavy-Ion Medical Accelerator in Chiba(HIMAC)", Particle Accelerators, 33, pp.147-152, 1990.

(2) T. Kohno *et al.*, "Control System for HIMAC Injector", Proc. of 7th Symp. on Accel. Sci. & Technol., RCNP, Osaka, Japan, 1989, pp.246-248.

(3) M. Kanazawa *et al.*, "RF Control for HIMAC Synchrotron", in these proceedings.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S04SRS01

Digital Control of the Superconducting Cavities for the LEP Energy Upgrade

G. Cavallari and E. Ciapala
CERN, 1211 Geneva 23, Switzerland

Abstract

The superconducting (SC) cavities for the LEP200 energy upgrade will be installed in units of 16 as for the present copper cavity system. Similar equipment will be used for RF power generation and distribution, for the low-level RF system and for digital control. The SC cavities and their associated equipment however require different interface hardware and new control software. To simplify routine operation control of the SC cavity units is made to resemble as closely as possible that of the existing units. Specific controls for the SC cavities at the equipment level, the facilities available and the integration of the SC cavity units into the LEP RF control system are described.

I. INTRODUCTION

The RF system for LEP phase 1 consists of a total of 128 RF accelerating/storage cavity assemblies operating at 352 MHz, providing a total circumferal voltage of 400 MV. The cavities have been installed around interaction points 2 and 6 in the form of eight individual RF units. Each consists of 16 cavities, two 1 MW klystrons, DC high voltage power converter, low-level electronics and controls. For the LEP200 energy upgrade to 90 GeV, requiring 2000 MV, a further 192 superconducting (SC) cavities will be installed in 12 new RF units, two at each of points 2 and 6 and four at each of points 4 and 8. The RF frequency for the SC units is the same as for the copper units and as far as possible they will use identical equipment i.e. klystrons, power converter, low-level equipment and controls.

The SC cavities are housed in groups of four in a common cryostat to make up an "RF module". Initially only one klystron will be installed per unit but there is provision to add a second for higher beam intensities.

Control of the SC cavities and associated equipment is based on the same principles as for the copper cavities with the same type of interface equipment and software. To render overall operation as straightforward as possible differences between the two types of RF unit are taken into account by local software. Unlike the copper cavity units the SC units can be run in different configurations e.g. four, eight, 12 or 16 cavities and with either one or two klystrons per unit. The configuration must be taken into account by the local software. With two klystrons SC cavity units could be operated more effectively as two "sub-units" of eight cavities and one klystron sharing the common HV power supply. This option is allowed for in the hardware layout and software.

At present one unit with 12 SC cavities is operational in LEP and further units will be installed gradually up to the completion of the project, planned for the beginning of 1994.

II. CONTROLS AND INSTRUMENTATION FOR THE LEP SC CAVITIES

Within the RF unit the various pieces of equipment associated with each major element of the unit are grouped together and controlled by a G64 bus standard based 'Equipment Controller' (EC). As for the copper cavity units there is one EC for each SC cavity but for each RF module there is an additional EC for cryogenics data. The EC is of modular construction and maximum use is made of a small range of interface cards. Standard modules are used to interface the following equipment :

Cavity tuning systems,
RF power measurement,
RF window and helium gas return heating,
HOM coupler fundamental mode power measurement,
Helium gas pressure measurement,
Helium gas valve control,
Cryostat insulating vacuum,
Cavity vacuum.

Interlock protection systems exist to switch off cavity tuning, RF or the HV power converter in the event of a fault or unsafe condition. These systems are largely contained within the ECs in the form of standard interlock modules and G64 readout interfaces which provide fault status and record trip sequences. For the SC cavities a "beam dump" interlock has been added. In the event of very high helium gas pressure or low liquid helium level, RF is switched off in all RF units.

For temperature measurements in the cavity a dedicated module inside the cavity EC measures up to 32 temperatures from signals from Pt100 sensors at various critical points. Conversion of voltage levels to temperature values in degrees Kelvin is done by the EC. Independent hardware logic inside the module triggers RF or tuner interlocks in the event of a change outside preset levels stored in EPROM. Fault information is stored and can be read via the the G64 bus.

A dedicated module in the cryostat EC provides a readout of liquid helium level in the cryostat. This, together with the gas pressure readings and RF levels, is made available via hard wired links to the regulation system of the cryogenics plant.

The design of the above equipment and interface hardware has been finalised and the material for the 180 cavities yet to be installed is being manufactured in outside industry.

III. LOCAL CONTROL

The control layout for the equipment of the RF unit, situated in racks underground in the klystron gallery beside the LEP machine tunnel, is shown in Figure 1.

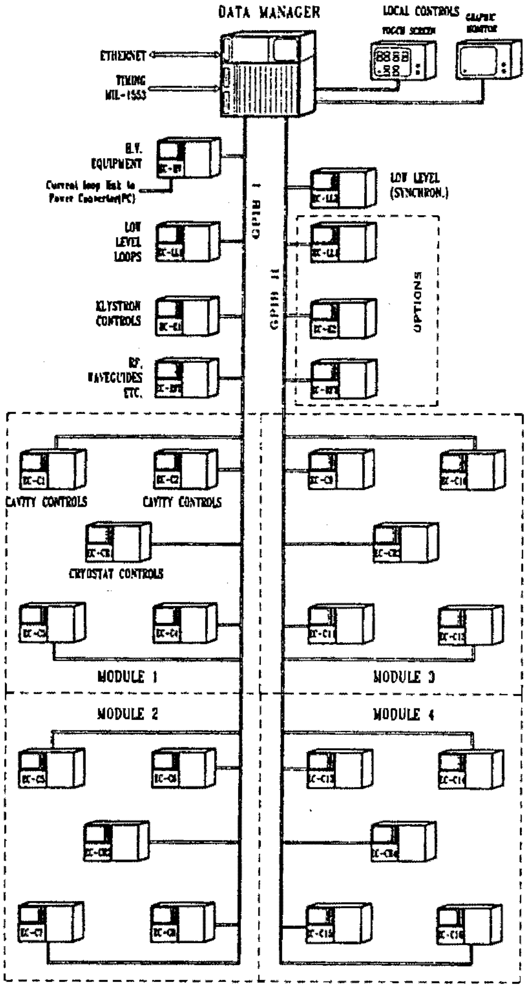


Figure 1 Control Layout within an SC Cavity RF Unit

An EC is dedicated to each major element of the unit i.e. HV equipment, klystron(s), low-level equipment, waveguide system, cavities and cryostats. For the SC units one or two klystrons can be installed, depending on available beam intensity. Second low-level and waveguide system ECs are allowed for to give independent RF operation of two [illegible]ht-cavity sub-units if two klystrons are used. Overall co[illegible] each RF unit is provided by a VME based "Data M[illegible] (DM) running a multitasking syst[illegible]. It communicates with the ECs via two local IEEE [illegible] (GPIB) buses.

The DM provides remote and local access to all equipment parameters via the ECs either singly or in groups. The DM equipment access functions have been modified for the case of the SC unit in order to cope with the various configuration possibilites. The low-level bus access routines make use of a preset array stored in battery-backed-up memory which indicates active ECs. The DM also executes control procedures involving access to various pieces of equipment, for example HV switch-on, RF switch-on, RF ramping and local surveillance.

For example the RF switch-on procedure for the SC cavity units involves the presetting of the various control loops, setting of initial klystron input drive level and cathode currents, clearing of any RF interlocks, setting the tuning to the correct mode and setting nominal tuning system reference values. After RF switch on klystron drive level and current are slowly increased to a given value of total forward power, sufficient to allow the cavity tuning systems to operate in the presence of LEP beam. As the cavities approach resonance they are slightly detuned by introducing a tuning reference offset to avoid producing full RF voltage and resulting large variation in synchrotron frequency. Once all cavities are in this state the voltage control loop is switched on, the tuning of all cavities returned to nominal and the starting RF voltage set.

While this and other procedures can be different from those of the copper cavity units they are called up in exactly the same way. All equipment and hardware dissimilarities are handled as far as possible by the DM and the ECs.

IV. OVERALL CONTROL OF THE LEP RF SYSTEM INCLUDING SC CAVITY UNITS

The DMs at each point equipped with RF are interconnected by an Ethernet segment and to the general LEP control system Token Rings by bridges.

The principles of operation for the SC units are exactly the same as for the copper cavity units. Direct remote access must be provided for all equipment functions in view of the large amount of equipment and the long distances from the control centre. At the same time maximum use is made of local intelligence at the levels of the DMs and ECs to reduce the amount of accesses required for standard operational procedures, such as switch on and setting up the RF voltage ramp for acceleration. These procedures are initiated by simple commands, the same for both SC and copper cavity units, and are normally carried out simultaneously in all units.

The Apollo workstations used for LEP operation in the Prevessin Control Room (PCR) communicate with the DMs and ECs over the network and local buses. Network communications are based on standard TCP/IP socket library routines[2]. Simple protocols based on command response using strings of ASCII characters are used throughout. The command formats sent resemble their corresponding equivalent "C" language functions with additional information denoting the unit and equipment EC to be accessed. The equipment access functions are implemented as simple macro definitions which invoke command response and appropriate data conversion procedures.

For example in a local DM program the "C" language function "s_valve(C9,3,40.0)" will cause the string "s_valve(3,40.0)" to be sent to the EC of cavity 9 requesting it to set helium gas valve 3 to 40 per cent of its range. The

EC carries out the action and returns a reply in string form e.g. "OK" for successful completion. Similarly the corresponding function at the level of the PCR workstation, "LRFSetValve(LRF_233,C9,3,40.0)", causes the string command "C9_RS,0_0,s_valve(3,40.0)" to arrive at the DM of LEP RF unit 233. The command format is compatible with the LEP standard command format intended for communication over the MIL-1553 bus : C9 is the device to be acted on in member 233 of family LRF and RS denotes that a reply in string form is expected. The reply is always in string form and when a numerical reply is to be returned it is converted at the source, the appropriate function being part of the macro definition.

This method first of all allows simple direct command response for remote access to equipment. This low-level access is essential in initial commissioning and has proved to be of value later on for troubleshooting. The command format reflects the logical path to the equipment and the function itself can be clearly understood and easily remembered by the equipment specialist. The underline character in the example given is included in all commands which change equipment states or values to permit a simple equipment protection system.

Secondly, the method of implementing equipment functions as macros based on the same simple direct command/response communication protocols allows straightforward build-up of more complicated software, both at the levels of the DM and the PCR workstations. The total number of functions defined both for the DM and for the PCR workstation is over 300, of which around 50 are specifically related to SC cavity control and data acquisition. The initiation and monitoring of DM procedures is done in the same way, in this case the equipment accessed is the DM itself.

PCR application programs for overall RF operation are based largely on the use of these DM procedures. They are essentially data driven. Use is made of a table called the RF current data set (CDS) which is set up by the operator to indicate to the program which units are to be acted on. The table is stored in the standard Table File System (TFS) format used for LEP applications software data. The RF CDS also contains data such as HV settings and available RF voltages for the various RF units. This means that one piece of application software can handle both SC and copper cavity RF units without the type of each unit having to be checked.

For the copper cavity units a large amount of software, based on interactive graphics packages, has been provided for the RF equipment specialists to show the detailed state of operation of the various pieces of equipment. The preparation of equivalent software to handle SC cavity and cryostat equipment is in progress.

V. SURVEILLANCE AND DATA LOGGING

In the same way as for the copper cavity RF units a DM background local surveillance program gathers the most important states, settings and readings from the SC cavities and other equipment in the RF unit. The data is stored in an operating system "data module" in the form of a structure. The unit configuration must be taken into account by the surveillance program and it deduces this from the contents of the active EC array. The data structure contains approximately 300 integers, strings and floating point values and is updated every 15 seconds. The information is also stored in a cyclic buffer in the local memory of the DM. At present up to one hour of data can be stored, but this will be increased by adding more memory to the DMs. The structure is transferred to the control room on request using TCP/IP protocols directly over Ethernet and Token Ring. It is used for overall logging of RF system performance and also for a permanently updating PCR display of RF system status.

The DM also carries out logging of parameters of special interest for cryogenics. Helium levels and pressures, RF powers and cavity temperatures are measured every 15 minutes by a background program and stored in ASCII form on local hard disc in TFS format. These files can be read remotely using Telnet or copied using FTP.

VI. OPERATIONAL EXPERIENCE AND CONCLUSIONS

The incorporation of the first SC cavity RF unit and its operation with the rest of the RF system has proved straightforward. The SC cavity unit is now routinely used during the running of LEP and its operation is treated in the same way as the copper units. As in the case of the copper cavity units the response time overall of the control system is adequate and its reliability good.

Whilst the software facilities for PCR operation are as complete as those for the copper cavity units, those for the presentation of collective data and detailed equipment states for the specialist await completion. In addition initial running-in experience has shown a need for more rapid monitoring, for example to help in the diagnosis of certain faults or when *in-situ* conditioning has to be carried out. The possibility of logging helium gas pressure and RF level for example and the detection of transients is envisaged. Special instrumentation, connected via the GPIB buses, to measure such parameters will be installed.

The gradual introduction of more SC cavity units will require practically no modifications to existing operational procedures. The control system hardware is finalised and in production and software facilities will be expanded as LEP200 installation progresses.

VII. REFERENCES

[1] E. Ciapala and P. Collier, "A Dedicated Multi-Process Controller for a LEP RF Unit", in Proceedings of the 1989 Particle Accelerator Conference, Chicago, March, 1989, pp. 1583-1585.

[2] E. Ciapala, P. Collier and P. Lienard, "Use of Ethernet and TCP/IP Socket Library Routines for Data Acquisition and Control in the LEP RF System", Contribution to the the 1991 Particle Accelerator Conference, San Francisco, May, 1991.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S04SRS02

A PC BASED CONTROL SYSTEM FOR THE CERN ISOLDE SEPARATORS

R. Billinge, A. Bret, I. Deloose, A. Pace and G. Shering.
CERN, PS Division
CH-1211 Geneva 23, Switzerland

Abstract

The control system of the two isotope separators of CERN, named ISOLDE, is being completely redesigned with the goal of having a flexible, high performance and inexpensive system. A new architecture that makes heavy use of the commercial software and hardware available for the huge Personal Computer (PC) market is being implemented on the 1700 geographically distributed control channels of the separators. 8 MS-DOS™ i386-based PCs with about 80 acquisition / control boards are used to access the equipments while 3 other PCs running Microsoft Windows™ and Microsoft Excel™ are used as consoles, the whole through a Novell™ Local Area Network with a PC Disk Server used as a database. This paper describes the interesting solutions found and discusses the reduced programming workload and costs that are expected to build the system before the start of the separators in March 1992.

I THE ISOLDE PROJECT

The ISOLDE project consists of the move of CERN's Isotope Separators and their experimental area from the recently de-commissioned Synchrocyclotron to a beamline served by the Booster Synchrotron [1] [2]. A new control system was required.

Traditionally, control systems for accelerators have been designed based on specified functionality, and the hardware & software tailored to optimize potential utility. Frequently however, this results in 'home-grown' products which remain incomplete and are overtaken by the rapid advances of the massive industrial base of commercial products.

The ISOLDE Project was taken as an opportunity to explore the extreme opposite approach for the control system. Namely to use 'market-leader' commercial software & hardware available for the huge PC market, with in-house development limited to the necessary software & hardware interconnects. This represents an experiment in providing an inexpensive, user friendly control system, requiring a minimum of manpower both for the implementation & for subsequent maintenance.

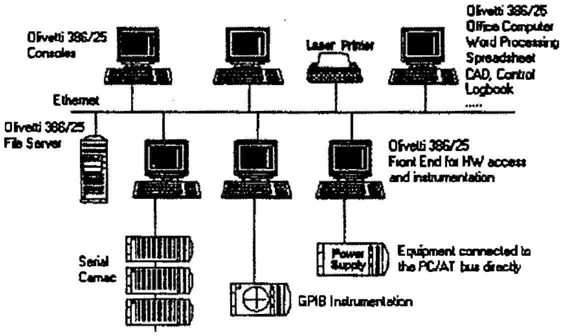
II THE CONTROL SYSTEM ARCHITECTURE

The new architecture [3] [4] [5] being installed reuses the old camac hardware while allowing an evolution of the system towards modern solutions.

Computers, Network and Control boards

The Isolde Control system is simple. It has Olivetti personal computers (PC) at all levels connected using the general purpose Ethernet network available side-wide.

As console in the control room, Olivetti 386/25 are used. i486 based PCs may be introduced next year. The console computers are equipped with 21 inch monitors providing a graphic resolution of 1024 x 768 pixels on 16 colors. Apart from high resolution monitors, the console computers are identical and fully compatible with the several hundred PCs available in the offices as local workstations [6].



The Isolde control system architecture

The personal computers connected to the equipment are called *Front End Computers* (FEC) and are also Olivetti 386/25. The performance of these computers is entirely satisfactory and 80286 PCs have enough CPU power to drive the equipment. In fact for some FEC applications, old IBM AT computers are used, recuperated from the initial Large Electron Positron Collider (LEP) front end controls, now replaced by faster Olivetti PCs. The Front End PCs are identical in configuration to the Office PCs except that they have additional boards for control.

The kind of control/acquisition boards supported in this architecture are:

- CAMAC, to allow the recuperation of all the hardware in the existing control system.
- GPIB to drive sophisticated instrumentation
- Industry Standard Architecture (ISA), alias PC/AT, boards that plugs into the PC directly or in ad hoc extension chassis.

A wide offer of this last type of boards exists and it has, for the moment, being restricted to Analogue to Digital converters (ADC), Digital to Analogue (DAC), Digital Input and Output (DIO), timer interrupts and external interrupt boards and RS232.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S04SRS02

Operating Systems and File Servers

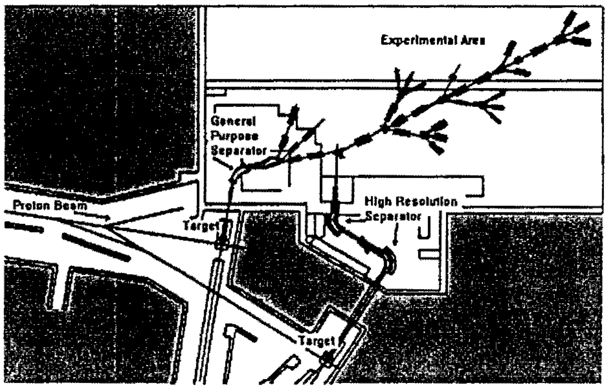
The whole system is built on a commercial PC network, Novell Netware™/386. This network provides a shared file system, shared printers, peer to peer and peer to server communications that are used in the control system to share databases and to share the hardware equipment between different consoles. The FECs are in fact just seen from the consoles as *equipment servers* to which client requests can be sent.

These FEC, alias hardware servers, are running MS DOS and the Nodal for DOS program. The Nodal for DOS program is an environment that provides:

- An equipment server to all consoles. The built-in network listener allows the consoles to access the equipment attached to the FEC.
- A local nodal interpreter that allows access to the equipment from the FEC. This is used mainly by the equipment engineers to test modules locally.
- Local alarm treatment

The Netware file system can be mounted also from Macintosh via Appletalk as an Appleshare file server and from Unix workstation via TCP/IP as a NFS server. All the Isolde databases or documentations in Word™ or Excel™ format can then be retrieved.

Numbers, manpower and financial resources



The CERN Isolde Facility

To quantify the dimension of the project, the control system comprises: 2 Faraday cage 60 Kv power supplies, 6 Extraction electrodes, 27 Faraday Cups, 5 Slits, 5 Lens Collimators, 15 Beam Scanners, 5 Wire grids, 6 Thermocouples, 18 Target power supplies in 60 Kv faraday cage, 9 Beam gates, 3 Separator Magnets with gauss meters, 58 Quadrupoles, 25 Steering Quadrupoles, 8 Kickers, 7 Benders, 4 Correction Plates, 3 Multipoles, 5 Deflectors and the Vacuum system. This gives roughly 300 devices (elements) and 1700 analog or digital wires (control channels) coming into the control system on three buses (CAMAC, GPIB, ISA).

More than 80 ISA (PC/AT) boards are being installed in different PCs or PC expansion chassis. 8 FECs are necessary, controlled by 3 consoles in the control room (or any office PC).

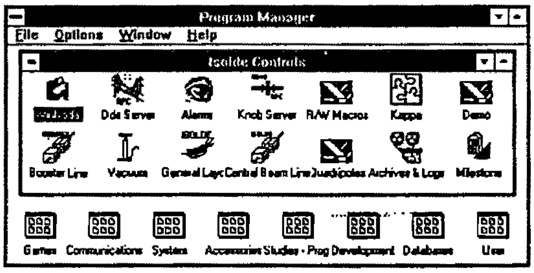
The manpower necessary to design and install the whole system is about 3 man*year.

Costs are equally divided between acquisition boards and computers. About 100 KCHF have been invested in PCs and the same amount in ISA boards. Wiring costs, 50 KCHF, are not included.

III BASIC TOOLS

Graphic User Interface (GUI)

The graphic user interface is Microsoft Windows™ and it provides a common access to all applications. The shell to start the different applications is the default *Windows Program Manager*.



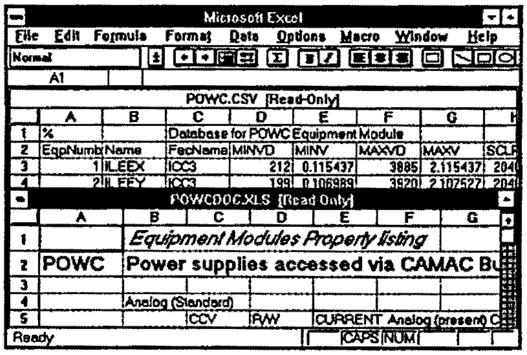
The program manager - Icons and menus oriented shell

Documentation

All documentation is produced using Microsoft Word™ for Windows, and is available on-line on the Isolde Server.

Databases

Databases are built using Microsoft Excel™. Database are stored in the native Excel format (BIFF) or in *Comma Separated Value* (CSV) format when they must be accessible from C or Nodal programs. The system is entirely database driven using Excel.



POWC.CSV [Read-Only]

	A	B	C	D	E	F	G	H
1	%		Database for POWC Equipment Module					
2	EqpNumb	Name	FecName	MINVD	MINV	MAXVD	MAXV	SCL
3	1	ILEEX	ICC3	212	0.115437	3885	2.115437	204
4	2	IL.EEY	ICC3	199	0.106989	3920	2.107527	204

POWCDOC.XLS [Read Only]

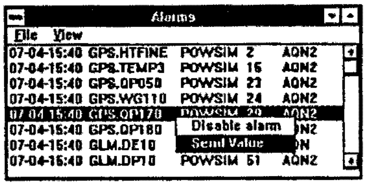
	A	B	C	D	E	F	G
1		*Equipment Modules Property listing*					
2	POWC	Power supplies accessed via CAMAC Bu					
3							
4		Analog (Standard)					
5			CCV	R/W	CURRENT Analog (present) C		

On-line databases

Alarms

An alarm process is running in all Front End Computers and systematically checks for inconsistencies in the element status or in the difference between acquisition and control values.

The alarm process has a list of active alarms that are polled by the alarm program running in the console. The Alarm program in the console is normally minimized in a green icon that becomes red when there are active alarms.



```
07-04-15:40 GPS.HTFINE  POWSIM 2   AQN2
07-04-15:40 GPS.TEMP3   POWSIM 16  AQN2
07-04-15:40 GPS.QP050   POWSIM 23  AQN2
07-04-15:40 GPS.WG110   POWSIM 24  AQN2
07 04 15:40 GPS.QP170   POWSIM 20  AQN2
07-04-15:40 GPS.QP180              )N2
07-04-15:40 GLM.DE10               )N
07-04-15:40 GLM.DP10    POWSIM 51  AQN2
```

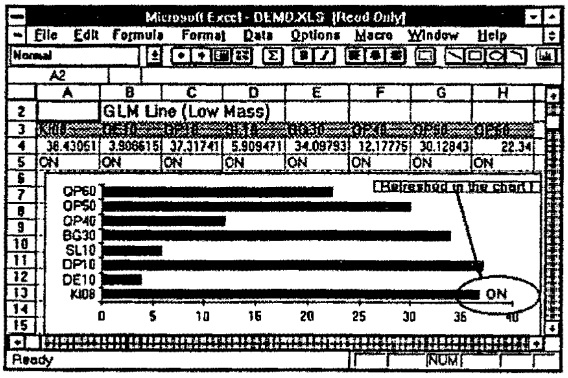
The Alarm eye

Applications

Three environment are available to develop applications. The first is Windows Nodal that provides a Nodal language interpreter with graphics capabilities that emulates the existing Isolde console.

Using Nodal for Windows it is possible to run all old applications developed in the Isolde history. The existing isolde console is made of a keyboard, a track ball, an alphanumeric display, a graphic display, two hardware knobs and a touch panel. Nodal for Windows emulates all these peripherals and permits reuse of the old software (see next section).

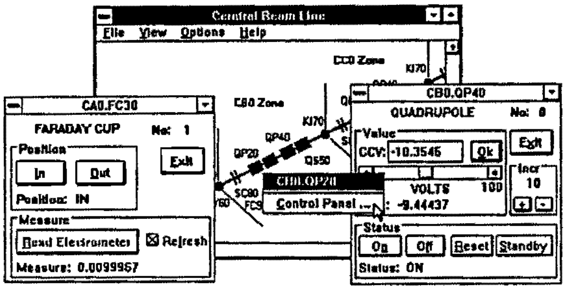
The second environment where control applications can be developed is in any application that supports Windows' dynamic data exchange (DDE). Several high level applications with programming facility could be used, for example Actor™, Toolbook™, Microsoft Word™ and Microsoft Visual Basic™. The recommended tool of this level is Microsoft Excel™. The Spreadsheet facility can be used to correlate physical quantities to control parameters and vice versa.



	A	B	C	D	E	F	G	H
2		GLM Line (Low Mass)						
3	KI08	DE10	DP10	SL10	BG30	QP40	QP50	QP60
4	38.43061	3.908615	37.31741	5.909471	34.09793	12.17775	30.12843	22.34
5	ON	ON	ON	ON	ON	ON	ON	ON

An Excel based application

The third environment where control applications can be produced is the Microsoft Windows Software Development Kit™ (SDK). Any application can be written in this environment and high performance and acquisition rates can be achieved.



No: 1
Position: IN
Measure: 0.0099967
No: 8
CCV: -10.3546
VOLTS 100
: -9.44437
Incr 10
Status: ON

An SDK application

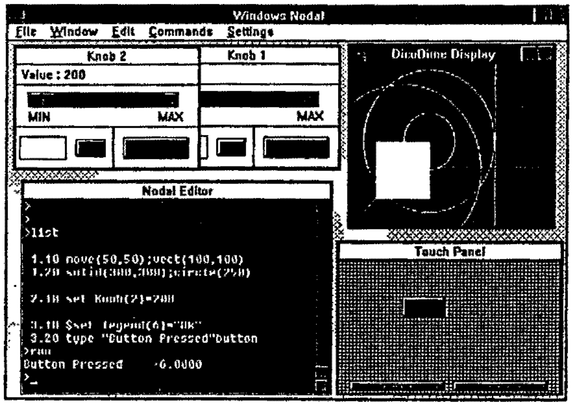
Security

Any application accessing the hardware must be attached to the Isolde Disk server where the client applications are stored. To attach a Netware file server a username and an encrypted password must be provided. In this way the Isolde Control System inherits all the security features that the Netware environment provides that are widely used by banks and other commercial Networks. In particular, it is possible to account the usage, to restrict the hours or days of the week a particular user can attach, to limit concurrent connections, force systematic password changes and several other features.

IV DEVELOPMENT TOOLS

Windows Nodal

The Nodal for Windows interpreter allows to run all existing Isolde software and permits the development of new applications. Interaction with the user is done using the two knobs and the touch panel. Data are presented in the alphanumeric and graphic displays.



Value : 200

```
>
>
>list

 1.10 move(50,50);vect(100,100)
 1.20 setin(300,300);circle(250)

 2.10 set Knob(2)=200

 3.10 $set legend(6)="Ok"
 3.20 type "Button Pressed"button
>run
Button Pressed    -6.0000
>_
```

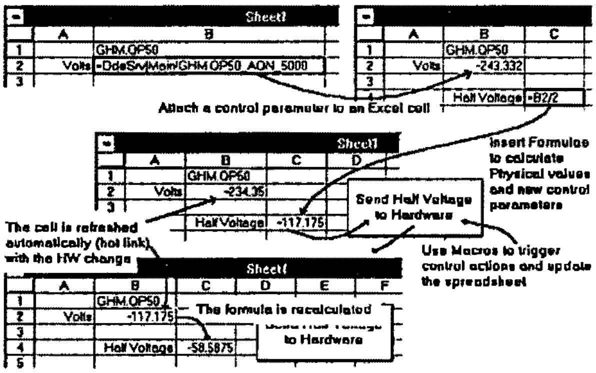
Nodal for Windows

Programming in this environment does not require any particular skill beside having read the Nodal manual.

Microsoft Excel™

Excel in addition to its database facility, provides spreadsheets, charting, database and advanced macro programming facilities.

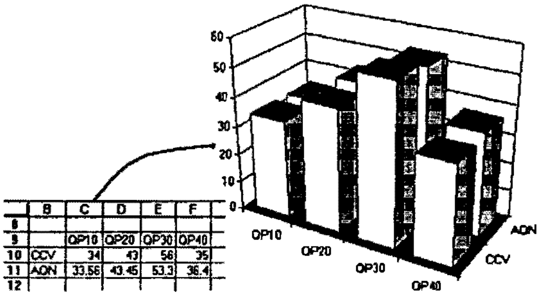
The spreadsheet facility provides the possibility of having hot-links, i.e. dynamic links between a cell in a spreadsheet and a particular property on a particular equipment, for example the acquisition of the current in a magnet. The spreadsheet cell is updated automatically whenever the physical acquired value changes. Formulae can be used to calculate from control parameters acquired with hot-links any physical quantity related to the parameter. All calculated values are, of course, also automatically updated when any quantity in the control system changes.



Example of application development using Excel

Basic macros, usually written automatically by the Excel Macro Recorder, allows to execute complicated algorithms. When attached to push buttons on the spreadsheet, macros allow to trigger complicated measurements, start optimization processes or to write calculated values to the hardware.

The Excel charting facility is used to produce graphics for analog presentation. Charts are normally also dynamically linked to the hardware properties and dynamically updated in real time.



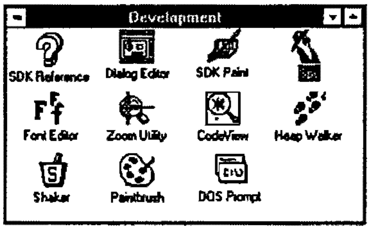
a dynamically refreshed chart

An Excel course for beginners is enough to allow a physicist to start building the applications himself.

Advanced Excel features can be used to produce applications with their own pull down menus and dialog boxes. The Excel Dialog Editor can be used for this purpose.

Windows Software Development Kit™ (SDK)

There are different tools available to design Windows applications, for example, the Dialog, font and Bitmap Editors, or the CodeView Debugger for Windows.



The SDK tools as example,

Writing applications using the SDK gives very high performances but a good knowledge of the C programming language and of the Windows architecture is necessary.

Microsoft Quick C™

The major tool used to develop Equipment modules is Microsoft Quick C. Beside providing an interactive Edit-Compile-Link-Debug environment as if C was interpreted, it has also on-line the complete C language reference. This context sensitive hypertext documentation provides also ready to run examples of all C library functions.

V CONCLUSIONS

Simplicity

Using the same type of computer (ISA PC compatible) at all levels of the control system leads to uniformity and simplicity. This advantage is reinforced by using the same operating system as the office machines.

All the persons involved in the design of this control system master it completely and are able to solve any problem at any level, front end or console. This global view on an uniform system reduces the skill necessary to cope with different parts of the system, because the approach is always the same, data driven from the Excel Database.

Development time

The use of well tested, market leader products (hardware and software) has reduced the development to a strict minimum. The very few necessary development done with the available tools have shown that:

- Development is very quick because of the user friendliness of the tools and because the SDK, the Microsoft C and QuickC compilers offer

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S04SRS02

incremental link and compilation options and run time dynamic link.

- Informatic skills are not necessary: Physicists and engineers can write their own programs after very little training.

In particular, the time to develop the application programs, normally the stumbling block in most control system is very significantly reduced: for example, it does not take more than 10 minutes to build a simple application using Excel to display graphically several parameters acquired from the separator.

Integration

The Isolde control system, although completely independent in case of problems, is transparently connected to the office network and all its services are available.

Beside providing an access to the Isolde control system from nearly 1600 points at CERN (the number of PCs installed), it is in the user culture by providing the same user interface to control applications as to any administrative or scientific applications. It is of course possible to Copy/Cut/Paste between any applications, for example, the Excel chart acquiring the status of the Isolde magnets can be pasted into Microsoft Word to produce a written report. Development of control application from any office is possible without the installation of additional hardware.

All A3/A4 laser printers (including the color ones), plotters, scanners, disks, software and data of the office network are available, including the support.

By relying on this infrastructure, the information can be easily distributed cern-wide as Statistics and/or electronic Logbook, as it is done now for the PS accelerator complex. As the file base is shareable between different platforms and operating systems (Macintosh, Windows, Unix), all data in Excel format can be retrieved from any user on the site.

Speed

The benchmarks done with the Olivetti 386/25 computers have shown that these computers are fast enough to control large processes where dozens of parameters from different FECs are involved. The margin can be enlarged in this area by using i486 based 33 MHz machines that are much faster.

The speed of the control system is often limited by the network used. The Novell network based on ethernet give us a performance of more than 100 Kbytes/second transfer rate on the busy CERN ethernet. A typical Remote Procedure Call with a search in the element database to find which FEC and which equipment module to call takes less that 30 milliseconds.

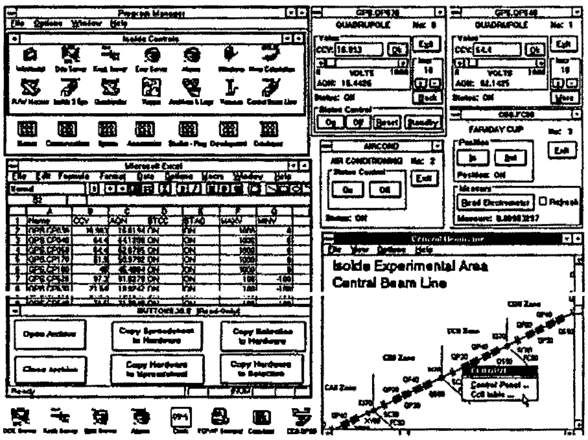
Cost

The market of Personal Computer in the world is roughly of 80 Millions units and with a strong competition between manufacturers. The PC production is more than 10 Millions/unit per year with performances increasing by a factor of 2 every year.

It' s a world wide standard, with binary compatibility, with major European involvement such as Olivetti, Philips, Bull and Siemens. A chassis, made in Europe, to control 48 power supplies (48 ADCs, 48 DACs, 288 digital I/O) cost about 12 KCHF. The cost per control channel is 9 CHF per digital bit, 55 CHF per ADC channel and 155 CHF per DAC channel.

Limits

There are no limits to the numbers of control channels the system can handle. If more are necessary, just add more FEC and more consoles.



A typical console screen

VI REFERENCES

[1] D.J. Simon, "Transfer of Isolde to the PS Booster", CERN PS/PA-EP Note 90-13

[2] B.W. Allardyce et al, "Isolde at the PS Booster", to be presented at the European Particle Accelerator Conference, Berlin, March 1992

[3] O.C. Jonsson et al, "The Control System of the CERN-Isolde on-line Mass-Separator Facility", EMIS-12, Sendai, Japan, September 1991. To be published in Nucl. Instr. and Meth. B

[4] A. Bret at al, "The Isolde Control System", CERN PS/OP/Note 91-27

[5] A. Bret at al, "The Isolde Control System Reference", CERN PS/OP/Note 91-17

[6] A. Pace, "Office Local Area Network at the PS Division", CERN PS/91-45 (OP)

STATUS OF THE CONTROL AND BEAM DIAGNOSTIC SYSTEMS OF THE CRYRING PROJECT

J.Starker and M.Engström for the CRYRING project group
Manne Siegbahn Institute of Physics
S-104 05 Stockholm, Sweden

Abstract—CRYRING is a facility for research in atomic, molecular and nuclear physics. It uses a cryogenic electron beam ion source, CRYSIS, together with an RFQ linear accelerator as injector into a synchrotron/storage ring for very highly charged, heavy ions. The first circulating beam was achieved in december 1990. The status of the systems for control and beam diagnostics are described.

INTRODUCTION

The CRYRING project [1] is centered around a synchrotron/storage ring of maximum rigidity 1.44 Tm, corresponding to an energy of 24 MeV per nucleon at a charge-to-mass ratio $q/A = 0.5$. It is mainly intended for highly charged, heavy ions produced by an electron-beam ion source (CRYSIS).

Light atomic or molecular ions can also be injected from a small plasmatron source (MINIS). Ions from the ion sources are accelerated electrostatically to 10 keV per nucleon and transported to a radiofrequency-quadrupole linear accelerator (RFQ) which brings them to 300 keV per nucleon. The ions are inflected electrostatically into the ring where they are accelerated using a driven drift tube. The stored ions will be cooled by an electron cooler. Fig. 1 shows a layout of the CRYRING facility.

The control system is based upon the LEAR (Low Energy Antiproton Ring) control system at CERN [2]. The principles of the system, the main part of the software and some parts of the actual hardware implementation are copied from the LEAR system. A substantial amount of development work has nevertheless been put into the CRYRING control system in order to adapt it to our operational needs which are partly different from the ones at LEAR.

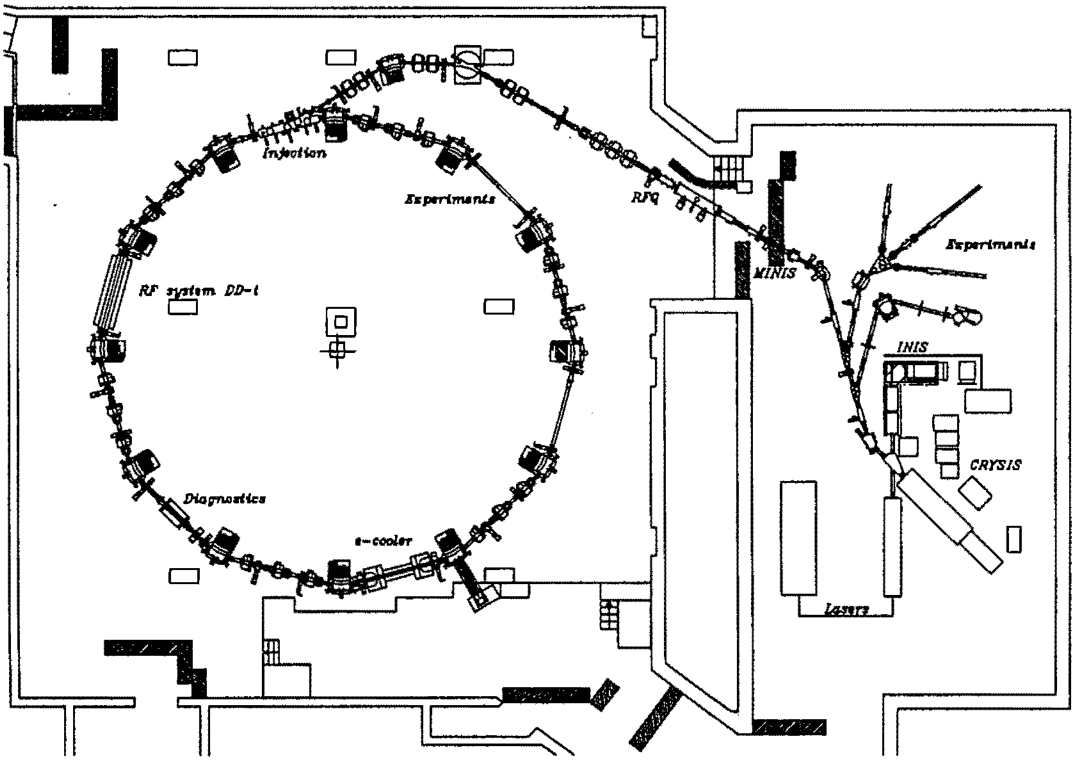


Fig 1 CRYRING layout.

CRYRING is equipped with different diagnostic elements to measure the low energy ion beam profile and current in the injection lines, to monitor the beam properties in the ring and to control the frequency of the accelerator structure keeping the beam centered in the beam tube.

THE CONTROL SYSTEM

The control system is based upon the LEAR control system at CERN. The architecture of the system, as well as the main part of the software and some parts of the actual hardware implementation are copied from the LEAR system. The development work in the CRYRING system has mainly been done in two areas. One is the low-level interfacing to our accelerator equipment using microprocessor systems with software written to allow local control for test purposes and thus make trouble-shooting easier. The second is the adaption of the software in the main computer to the actual hardware and to our operational needs which partly differ from the ones at LEAR. The system controls the whole CRYRING complex, that is the ion sources, the beamlines and the ring. The electron cooler, which will be installed this winter, will also be controlled by the same system. The general structure of the control system is shown in fig. 2.

Structure of the system

The main components of the systems are:

- The main computer, a PDP-11/73, with terminals and two operators consoles.
- A serial CAMAC loop for distribution of data and timing.
- A large number of G-64 microprocessor systems for interfacing to the machine equipment.

A piece of accelerator equipment connected to the system is called a parameter. Analog control signal output to parameters from interfaces in G-64 systems can be of two different types: static, controlled directly by the main computer, or ramped, controlled via programmable function generators in the G-64 systems. These function generators, called GFDs [3], have the shape of the function down-loaded from the main computer via the CAMAC serial loop. They can then be started, stopped, held and released synchronously with each other by pulses from the timing system. The timing system [4] consists of one master and a number of decoder modules, all sitting in CAMAC, interconnected with a timing distribution cable.

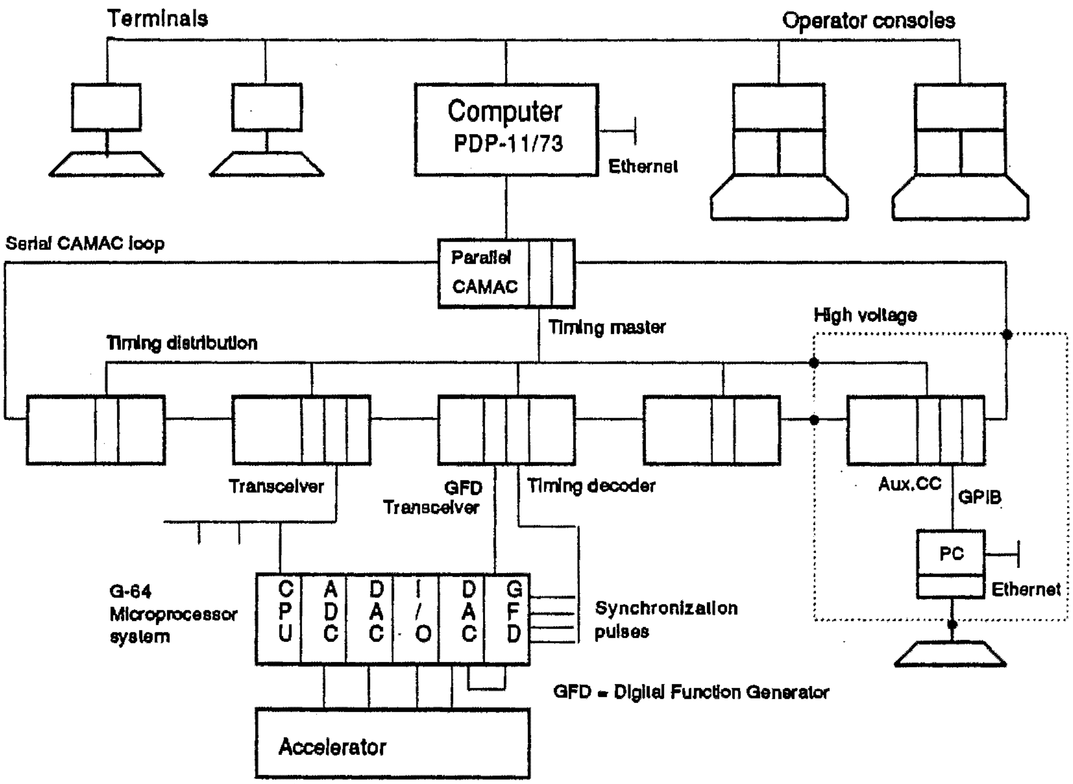


Fig 2 The control system.

The master and decoders are programmed via CAMAC by the computer to execute a predifined set of timing events, a machine cycle. Starting, stopping and repeating the machine cycle is done by external pulse inputs. This means that once the GFDs and the timing system have been loaded the cycling of the ramped parameters is done without need for intervention by the main computer.

The control of the CRYSIS ion source demands special flexibility and quick responses, as well as a more developed local control. These needs have been met by adding a PC that has access to the necessary hardware via an auxiliary crate controller in one of the CAMAC crates. This has requred an optical link between this PC itself and its keyboard, because the PC as well as the mentioned CAMAC crate is put on a 50 kV platform.

Status, November 1991:

The control system

- has been running since the first parts of the accelerator system were taken into operation in 1987.
- presently controls around 160 static and 20 ramped parameters. New parameters are successively added.
- allows control from both control room and equipment areas.

The G-64 systems

- control from 1 to 16 parameters each.
- have hardware and software tailored for their actual applications.
- can be equipped with terminals and run in local mode, thus making it easier to trace faults in accelerator hardware and in main computer software.

Development:

Some upgrades are being considered to improve performance and user interface. These developments are parts of today's control system at LEAR and would be logical updates to the CRYRING system:

- More computing power can be added by network linking to VAX computers at the institute.
- Workstations and PCs are considered as complement to the operators consoles. Operation with graphic presentation, also from equipment rooms and experimental areas, is desired.

DIAGNOSTICS

The beam intercepting devices in the transfer lines are Faraday cups for intensity measurements and strip detectors for beam profile and emittance measurements. Also, chromium doped Al_2O_3 plates viewed through TV-ca/-me/-ras are used here.

The control system moves these devices into/out of the beam, switches TV-cameras to different monitors and the read-out of the beam current to selected instruments.

The signals from strip detectors (each detector consists of 16 horizontal and 16 vertical strips) are amplified, multiplexed and sent to ADC's in a VME computer system placed in the control room, fig. 3.

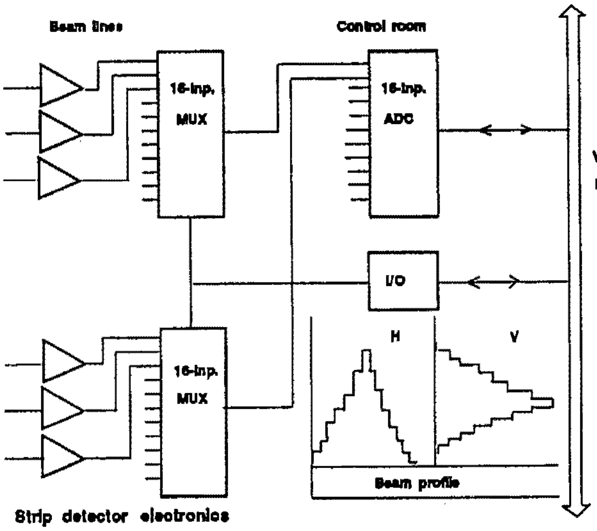


Fig 3 Strip detectors.

In the ring non-destructive measurements are performed using 9 horizontal and 9 vertical electrostatic pick-ups, a beam current transformer and a Schottky noise detector.

The signals from the pick-ups, fig. 4, can be processed by a fast peak-detection system [5] or by using synchronous rectifiers for low-bandwidth measurements. An even faster peak-detection system is now being installed on one pick-up, to allow for measurement over 128 consecutive turns, to study transient behavior of the beam [6].

A high resolution (300 nA) current transformer from Ber/-goz, for measuring the absolute value of the current of the unbunched beam, was installed. So far there has been problems with the signal-to-noise ratio where the noise has been observed to come mainly from the ring magnets . Work is going on to solve this problem. The electronics of the Schottky detector is shown in fig. 5.

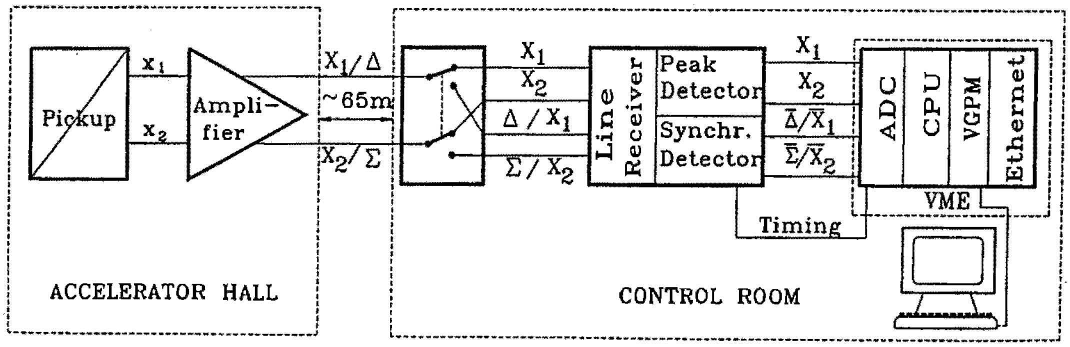


Fig 4 Pick-up detection system.

The input signals are amplified by high bandwidth (.01-50 MHz) amplifiers and the difference and sum signals are created in a passive circuit consisting of three power splitters. Frequency analysis of these signals can yield the q-value of betatron oscillations and the momentum spread of the beam, as indicated in the figure.

The same result, but much faster, can be obtained by processing the signals in a DSP consisting of a flash-ADC and a FFT processor. It is considered to include this type of module in the VME system. The VME computer is equipped with a GPIB controller which allows control and read-out of auxiliary instruments.

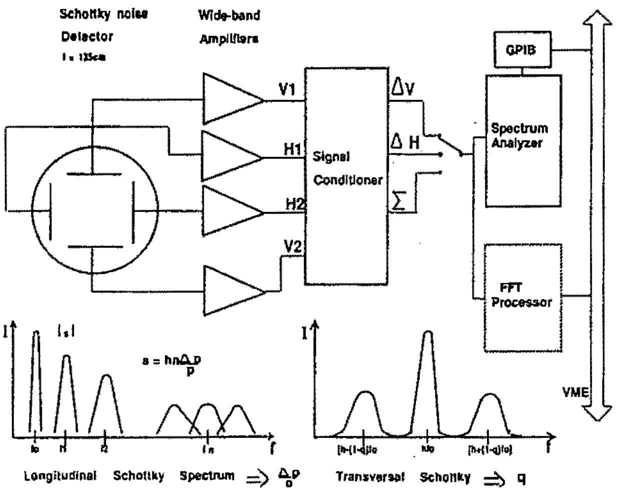


Fig 5 Schottky detector.

References

[1] C. J. Herrlander, K.-G. Rensfelt and J. Starker, "CRYRING—a heavy-ion synchrotron and storage ring," in *EPAC 88* Rome, June 7–11, 1988, p. 350; K.-G. Rensfelt, "Status of the CRYRING project," in *EPAC 90* Nice, June 12–16, 1990, p. 623; C. J. Herrlander, "CRYRING— a Low Energy Heavy Ion Facility", in *Cooler Rings and Their Applications*, Tokyo, November 5–8 1990, to be published.

[2] U. Tallgren et al., "The distributed computer control system for the CERN low energy antiproton ring (LEAR)," CERN PS/85-27 (LI) 6.5.1985.

[3] P. Lienard, "The LEAR GFD," CERN/PS (to be published).

[4] J. Knott, "The LEAR control timing system," CERN PS/LI/Note 83-12.

[5] S. Borsuk, K. Agehed, W. Klamra, and Th. Lindblad, "A peak detecting pulse height ADC system for beam diagnostic pick-up detectors in the CRYRING accelerator," *Nucl. Instrum. Methods*, vol. A284, p. 430, 1989.

[6] S. Borsuk, "The fast digital peak detector for the beam diagnostic system in the CRYRING accelerator," (to be published).

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S04SRS04

MAGNET TEST FACILITY CONTROL SYSTEM FOR SUPERCONDUCTING MAGNETS OF UNK

A.I.Agueev, V.N.Alferov, K.F.Guertsev, V.I.Gridassov,
A.A.Gussak, A.F.Dunaitsev, V.A.Krendelev, A.F.Lukyantsev,
V.M.Proshin, V.E.Solovyov, A.N.Sytin, E.A.Ustinov.

IHEP, PROTVINO, USSR.

1. INTRODUCTION

An UNK Magnet Test Facility (MTF) is being constructed to provide cryogenic, electrical and magnet tests of superconducting (SC) magnets of UNK. The main parts of it are:

- The cryogenic system consisting in its turn of the central liquefier, ten satellite refrigerators, two compressors, purification system and transfer lines. The central liquefier supplies the satellite refrigerators with liquid helium. The liquefier is manufactured according to the scheme incorporating precooling by liquid nitrogen, two turbine expanders and a wet expander.

- Four 8 KA, 24 V, ramped Power Supplies (PS) for cold testing of SC magnets, two 3 KA PS's for instrumentation testing and calibration.

- Test facility in its turn consisting of:

a) two dipoles and one quad benches for warm measurements;

b) eight dipoles and two quad benches for cold measurements;

c) two benches for instrumentation.

Relevant parameters and technique are given on table:

	Item	Parameter	Technique
1	Reference and calibration dipoles	Axis field	NMR - method in the central region and Hall - method at the end parts
		Multipoles	Rotating coils
2	Dipoles and quads measurements	Effective length Multipoles	Rotating coils NMR
		Field angle Magnetic axis	Stretched wires
		Dynamic multipoles	Stepwise coil rotation Measurement of transition process after dB/dt→0

The systems for warm measurements of SC coils and cold measurements of quads, including measurements of the magnetic axis location and alignment of the reference target, are being developped jointly with Saclay (France).

Total production rate of facility intended to be 2,4 SC magnets per day.

Such a complex of equipment requires a Control system which provides automatic monitoring, control of equipment, data aquisition and storage into files of magnets. Taking into account the difference between 3 pieces of an equipment (cryogenics, PS's and electrical/magnet measurement stations) Control system is designed as a mixture of 3 different subsystems with different phylosophy, but connected by LAN, sharing the same Host Computers, and based on the same hardware and computers.

2. HARDWARE CONFIGURATION

The hardware configuration is shown on Figure. PC/AT's were chosen as a suitable computers for real time control of groups of equipment and measuring benches.

PC's present the middle level of computers hierarchy. The upper level is formed by two Host Computers of DEC - family. All computers are connected with Ethernet.

The interface electronics is based on IHEP version of CAMAC. In cryogenics the groups of CAMAC crates communicate with PC's through RS-232 lines, connected to crate controllers. CAMAC crate houses:

- *intelligent crate controller with 16 bit, LSI-11-compatible processor;*
- *convertors for Allen Bradley thermometers;*
- *convertors for vacuummeters;*
- *12 bit ADC;*
- *I/O - registers;*
- *restart memory module after power failure/restore;*
- *thyristor and relay drive amplifiers for Motors and valves;*
- *intelligent (16 bit, LSI-11-compatible) module for 4 control loops, aimed to provide reliability of the Control system itself as well as the cryo-complex.*

Such a distributed processing means that all processors provide local control algorithms, gather local data pool of all related equipement, transmitting information to upper level only on request.

Control of PS's is provided with 16 bit computer, connected to CAMAC. Electrical/magnetic measurement station consists of 2 CAMAC crates, connected to PC, CAMAC crates house:

- *18 and 16 bit ADC, 20 bit DAC, timers, function generators for PS's control;*
- *amplifiers, filters, comparators for quench detection;*
- *NMR - convertors, voltage/frequency convertors, step motor drivers and so on for magnetic measurements.*

3. SOFTWARE

The purpose of software is to maintain functionality of all subsystems connected with production, testing and filing of magnets. In addition (independently of the serviced subsystems) software must maintain:

- *automation of programming that includes the programming of low-level computers;*
- *communication between all computers that includes loading and start-up of programmes on low level, data exchange, subsystems interface through a middle-level computer;*
- *filing of the produced superconducting magnets and preparation of all necessary documentation about them;*
- *support of the bank of programmes that maintain execution of the system tasks.*

All software may be devided into the system software and the application one.

The application software carries main functional load and includes algorithms of control, data acquisition, mathematical processing and representation of data that are individual for each technological subsystem. The software toolkit necessary for quick and qualitative writing of application programs is maintained by the system facilities.

The system software is all software that doesn't depend on the object of control and one that is common for various subsystems. It maintains the operational environment for application programs interfacing with operator and equipment and includes the following components:

- *operating systems of high- and middle-level computers and their utilities;*
- *systems of automation of programming;*
- *software for intercomputer communications (TCP/IP) that includes the communication of high-level computers with the computers of IHEP central complex;*
- *automation system data base and its utilities;*
- *the operational bank of superconducting magnets tests results;*
- *packages of subroutines for organization of the operator dialog with computers, format data conversion and data representation on the information representation equipment;*
- *multiprogram real-time monitor for low-level computers;*
- *the tests for system equipment components.*

The features of problems solved by high-level computers permit us to use the VAX/VMS operating system that maintains a parallel execution of some tasks in multiprogram and multiaccess modes. We want to use the mode of the RSX-11M operating environment emulation especially in the system of automation of programming of tasks for low-level microcomputers.

We have selected C and FORTRAN as the programming languages for high-level computers. The features of the programming of low-level computers are defined by two reasons: small amount of main storage (56 Kb) and the short response time requirements especially with respect to control programs. The first reason requires elaborate and accurate programming. Second one requires the use of specially developed monitors.

The middle-level computers may be used as the local console for subsystems that need a control.

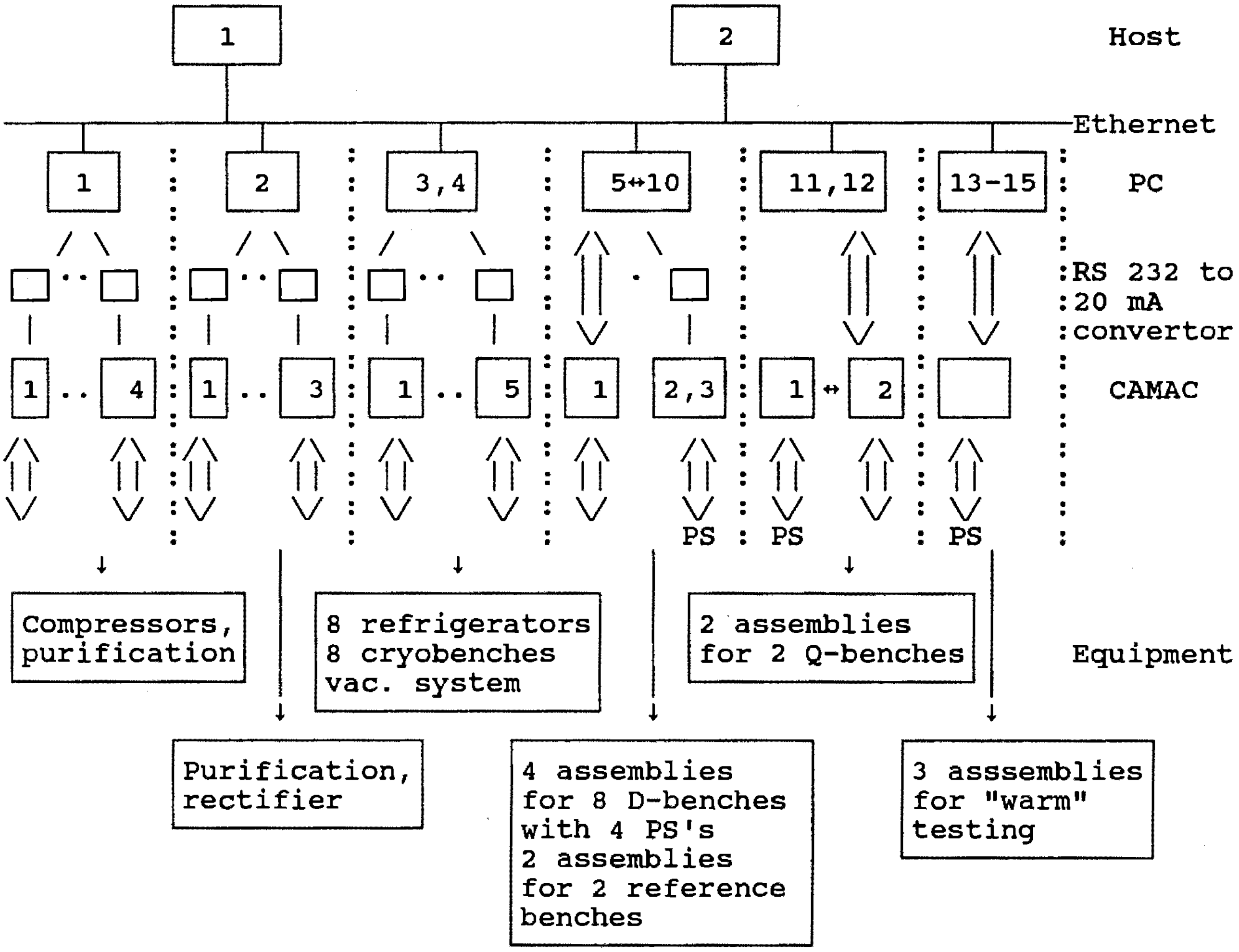
4. INTENTIONS

a) Analog electronics interlock is being constructed to protect the most important cryogenic equipment in case of Control system default.

b) Cooling of CAMAC crates with help of vortical air refrigerator will be tested.

5. ACKNOWLEADGEMENTS

The author list of this status paper contains only names of group leaders. Much more people really take part in the designing of the Control system.


1
2
Host
Ethernet
1
2
3,4
5↔10
11,12
13-15
PC
:RS 232 to
:20 mA
:convertor
1 .. 4
1 .. 3
1 .. 5
1
2,3
1 ↔ 2
CAMAC
PS
PS
PS
Compressors,
purification
8 refrigerators
8 cryobenches
vac. system
2 assemblies
for 2 Q-benches
Equipment
Purification,
rectifier
4 assemblies
for 8 D-benches
with 4 PS's
2 assemblies
for 2 reference
benches
3 asssemblies
for "warm"
testing

ISBN: 978-3-95450-254-7	ISSN: 2226-0358	doi:10.18429/JACoW-ICALEPCS1991-S04SRS05

BEAM EXTRACTION CONTROL SYSTEMS OF THE FAST-CYCLING SYNCHROTRON

A.Toumanian, N.Zapolski, V.Nickogosian, A.Ananian,
A.Kazarian, M.Khoetsian, A. Agababian, A.Matevosian
Yerevan Physics Institute, Armenia

Abstract

A compact system controlling the extraction of different beams (gamma, electron, synchrotron radiation) in single and simultaneous operation modes at high electromagnetic disturbances level based on using one computer of IBM PC/AT type is described.

Introduction

Physical research program at the Yerevan synchrotron pursues the realization of the experiments generally with the use of the slow extraction of primary and secondary beams in single and simultaneous operation modes at 4.5 GeV energy with the 4-8 μs magnetic field top. The most complicated process of the extraction, requiring the precision tuning of the beam extraction devices and not having analog in the world is the mode of simultaneous beam guidance to the two internal targets, one of which is a thin crystal, the other one is of thick tungsten and is put in the neighbouring focusing interval of the synchrotron. At the same time it is necessary to provide a significant decrease of the beam pass factor through the thin target by screening from the particles, once passed through it by means of the thin target [1].

Due to the developed and described below the control system of the synchrotron extraction devices it was managed to increase the ratio of the pick of coherent bremsstrahlung radiation from the thin crystal target to the amorphous part almost 2.5-3 times with keeping unchanged the common requirements to the extracted beam parameters, that is to say to the stable uniformity and duration of the extraction, effectiveness of the extraction and so on.

Secondary beam extraction of the Yerevan synchrotron is based on the local disturbance of the orbit with using the additional electromagnetic coils of the guiding magnetic field. At the beam guidance simultaneously onto two internal targets there is also used a system of changing the betatron oscillation frequencies of the circulating particles with the help of the lenses set on the orbit. To realize the slow extraction of the primary beams to the vicinity of the nonlinear resonance of the third order the conventional system of magnetic elements (quadrupole and sextupole lenses, septum and bending magnets) is used. Magnetic elements and additional coils of the electromagnet are supplied by the current pulses of the complicated shape, produced by the resonance forming lines with the use of the thyristor switches. The tuning of the form and amplitude of the current pulses is realized by means of the face control of thyristor switches with the use of the synchronizing pulses from the synchrotron timer device. The control of the current pulse form and the intensity changes during the beam extraction is carried out by many pickups.

1. Architecture and the control system construction principles

The first control system of the synchrotron extraction device was based on the control computers EC1010 and EC1011 (Videoton firm, Hungary) [2]. But the lack of reliability in their work and the relatively expensive maintenance showed the necessity of replacing them by the modern computers. The computer PC/AT was chosen for that. It determined the architecture of the control system from the one hand and from the other the requirements of reliability and flexibility at high level of the electromagnetic noises were satisfied by having an intensive information flow, a large number of the control parameters and so on. That's why a mixed 3-level architecture of the computing systems was chosen (fig.1).

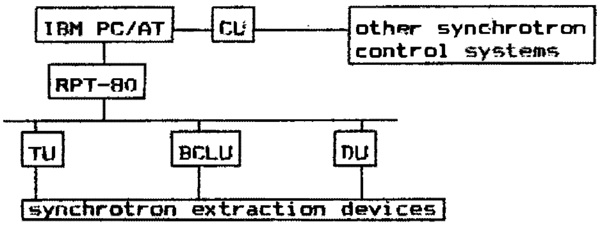


Fig. 1

A set of specialized microprocessor modules of KP580BM80 type, allowing to solve the following tasks was developed for operation at the low level:

- continuous measuring, tolerance parameters control and formation of the actions for controlling accelerator extraction devices;
- buffering, preliminary processing, information conversion and transmission to the computer of the higher level;
- synchronization of measurement and control processes with the synchrotron cycles.

The wish to minimize and get less expensive apparatours from the one hand and to achieve the sufficient universality of its functional possibilities from the other - was the main reason of the development of these modules and of not using the nucleus electronics apparatours.

The microcomputer RPT-80 (Hungary) used on the middle level with the processor of INTEL 8080 type runs system terminal functions in the separate control subsystems at its off line work and as a peripheral processor at the controlling through the higher level. In the first case it solves the user tasks providing a standard interface to all the modules of the low level and in the second one it solves the same problems as well as the other ones but under the control of the computer higher level.

At the higher level a personal computer IBM PC/AT is used, the main functions of which are the following:

- creation and maintenance of the parameter data base of the main operation modes of the beam extraction devices;
- realization o the local control algorithms with the feedback;
- statistic processing of measurement results at the normal system operation;
- information exchange with the other synchrotron control system.

For information exchange with the other synchrotron control systems, specially with rf systems, electromagnet supply system and others, the second serial port of IBM PC/AT is used, as well as a non-standard communication unit (CU).

1.1. Timer Unit (TU)

The timer unit is developed on the base of the microprocessors and is used for synchronization of all the extraction devices and equipment of physics-experimenters with the synchrotron acceleration cycles and carries out the following functions:

- control of the synchronization main pulse and its selection on the false signals background measurement and tolerance control of frequency; in case of mode violation of the main pulse forming timer unit automatically switches off the controlling of the extracted beam channels for elimination of the break-downs in the thyristor devices;
- program distribution of the synchronization main pulse in the devices of different extraction beam lines depending on the operator given sequence;
- time pulse delay formation in the given devices for running the phase control.

Main technical features

- pulse distribution channels number - 8;
- the range of the programmed pulse time delay is within 0.5 μs - 32 μs;
- pulse distribution periodicity in the beam lines is arbitrary - up to 256 cycles.

1.2. Beam lines control units (BLCU)

Eight units of the beam control lines are based on the unified microprocessors for control all the parameters of the magnet extraction devices and have the following functions:

- measuring with the help of the different ADC - the current control pulse in the magnets and beam intensity signal from the scintillator pickup and realization of their tolerance control (the number of sampling points at analog signals measurements is up to 256, sampling step is 55 µs, measurement accuracy is 12 digits);

- the phase control of the amplitude and shape of the control current in the magnets by means of the 6 control time intervals for each forming current pulse;
- measuring and tolerance control of time intervals between synchronization and controlling pulses (up to 16 time intervals);
- information exchange with the higher level computer.

1.3. Diagnostics unit of the relay signals (DU)

This unit realizes registration, control, diagnostics of the state signals of the "switch on - switch off" type (number of channels -64, block time response to the state change - not more than 100 µs).

1.4. Software

The low level microprocessor units software is based on the program-monitor, realizing the main cycle of unit operation and organizing communications with the subprograms as well as on the asynchronous lines, drivers including subroutines of the data exchange, control words receive determining equipment operation mode and status-words transmission programs of the corresponding unit.

Middle level software [3] realized in RPT-80 consists of command monitor, global control table and command table, input-output dispatcher of the driver external devices, manager of the asynchronous communication lines and the interrupt handlers.

Command monitor realizes the direct interaction with the operator through menu, which gives the list of all available tasks and the ways of access to them. Monitor also supports the global control table and the command table.

The information of the external device operation mode, the interrupt bytes and also the pointer to system area, which is given to each external device, are kept in the control table. Command table contains the addresses of the functional tasks.

The input-output dispatcher is created for a common access to different external devices.

Manager of asynchronous communication lines gives an alternative command input source which realizes the information exchange between the computers.

The described software is written in the assembly language in the CP/M OS environment and occupies 2 kB of ROM.

Conclusions

The created system realizes the following possibilities:

- continuous control of the extraction devices state;
- measuring and display of the current values of all the measured parameters in the digital and graphical form;
- monitoring the extraction beam quality and fault diagnostics;
- manual and automized control of the extraction devices through the computer while tuning the extraction modes and stabilization of their parameters.

More than one year operation of the system proved the reliability of its work and the convenient maintenance.

REFERENCES

1. A.R.Toumanian, Kh.A.Simonian, N.A.Zapolski. The way of getting the secondary beam in the high energy cyclic accelerators. (in Russian).
2. A.G.Agababian, and others. - The structure and organization of automized control subsystems by the beam extraction from the fast cycling synchrotron. Proc. of the II Sov.Union conf. on charged particles accel.V.1,p.125,Dubna,1989. (in Russian).
3. A.G.Agababian and others. Software subsystems control organization of the Yerevan synchrotron. Preprint YerPhy 1284(80)-90, Yerevan 90. (in Russian).

Instrumentation & Control System For PLS-IM-T 60 MeV LINAC

D.K.Liu, K.R.Yei, H.J.Cheng, L.P.Yang, C.Z.Chuang
W.Yue, P.Lu, Y.H.Su, Q.Lin, H.Huang
Institute of High Energy Physics
P.O.Box 918 Beijing

Abstract

The PLS*IMT* is a 60 MeV LINAC as a preinjector for 2 GeV LINAC of PLS project. The instrumentation and control system have been designed under the institutional collaboration between the IHEP (Beijing, China) and POSTECH (Pohang, Korea). So far, the I&C system are being set up nowadays at the POSTECH of Pohang. This paper describes its major characteristics and present status.

I. INTRODUCTION

The Concept Design Research(CDR) of PLS 60 MeV LINAC has been completed in 1989. The construction of PLS 60 MeV started in July, 1991. The accelerator column and electron gun have been installed earlier. The gun pulser has been tested with 3.5A 2ns pulse width with success and Modulator, microwave system and I&C system will be set up soon. The commissioning of whole system would be completed around the end of this year or next spring.

The I&C system of PLS 60MeV is a compact and complete hierarchical distributed control system. Therefore it is small system and it includes all of the essential control structure and various beam monitor, high speed electronics modules etc. for LINAC operation.

II. SYSTEM STRUCTURE

In a centralized control system, computer failure will cause a failure that will shut down the entire system. However, a distributed system is more costeffective and becomes easily modified.

According to the requirements of physics and our previous experience, and considering the entire budget, schedule of I&C of PLS 60MeV, we compared various structure of control system [1], and adopted the Intel BITBUS architecture. The major reasons are as follows:

* BITBUS distributed control system is a commercial product

* High performance microprocessor could be useful for local station.

* Powerful software support such as RMX286 and RMX51 are an excellent developing environment. The function that have to be explicitly coded can be greatly reduced by making system calls. A BITBUS drive can be run under the RMX286 which allows messages passing across the SBX interface on down the BITBUS network.

* More second source: We should consider the situation that developing this system is in China, and commissioning and maintenance is in Korea. So we must get these products easily from the market of both country.

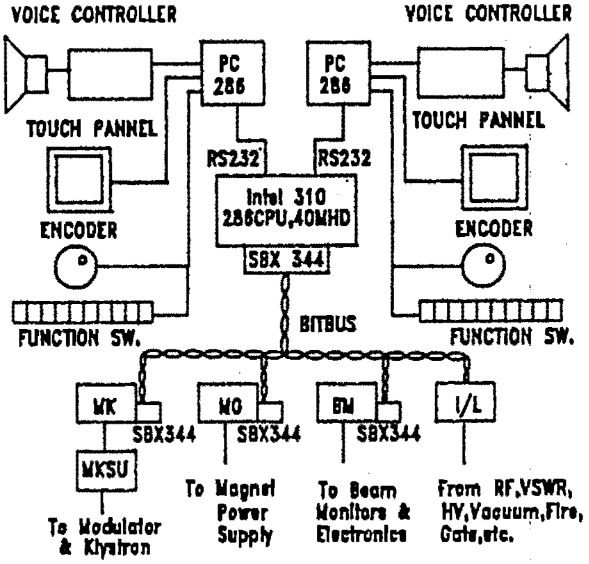


Fig.1 I&C archtecture for Preinjector of PLS

According to the considerations above, the system architecture is illustrated in Fig.1.

There are four stations linked with BITBUS network, each station has its own resource and tasks respectively. Those local stations are Modulator-Klystron Station(MK), Magnet power station(MG), Beam diagnostics station (BM) and Interlock station(IL).

In general, entire task are hierarchically managed. Each local station completes data acquisition and data control during the 5ms period. the details of MK local station will be discussed as example. Intel 310 (CPU 286, HD 40M) can be used for task management, data sort, data processing and BITBUS communication control. Two sets of industrial level console computer (CPU 286, HD 40M, RAM 2M) which can be used for humanmachine interaction.

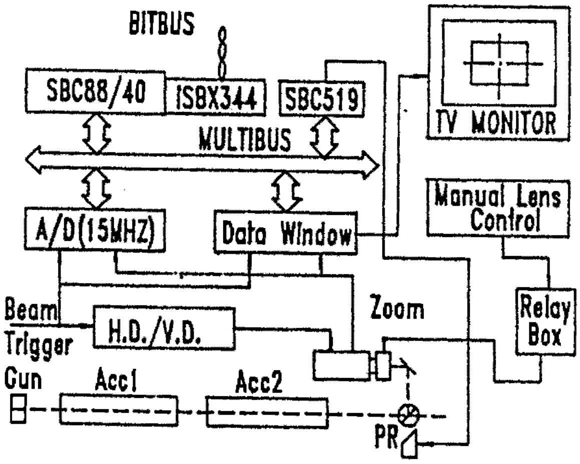


Fig.2 Beam Station Schematic

III. HARDWARE SYSTEM

A.Local station

Each local station has own iSBC 88/40 microprocessor and various I/O interface boards which are linked together on the Intel's MULTIBUS.

1.Modulator and Klystron Support Unit (MKSU) [2] is a powerful interface between MK station and Klystron & Modulator. It was developed by SLAC in 1985. and contains various interface circuit linked with intelligent PIOP CAMAC module. In order to keep this powerful function in our system, a dedicated bus adapted from MULTIBUS to MKSU bus have been developed successfully.

2. MG station

The PLS 60 MeV has 29 sets of power supplies to be monitored and controlled for serving the solenoid coil, steering coil, analyzing magnet and quadruple magnet. A digital remote control model using data modulator and demodulator with Manchester code has been adopted. By the way, a remote D/A controller could be mounted at the magnet power supply as close as possible.

3.BM station is designed for measuring various beam characteristics such as beam profile, beam energy spread and beam emittance. This is a multifunction image processing system, illustrated in Fig.2.[3]

It consists of a profile monitor, a video signal synchronizer, a high speed A/D converter with 15 MHz inserted into the Intel's MULTIBUS. The video signal of beam spot from the camera was transmitted to the TV monitor of control room. It is easy to correct the target haircrossing line using the movable data window by the computer control. The 4000 points signal could be collected in less than 4ms. After data processing, a beam profile distributed picture and three dimensional distribution will be shown to operator immediately.

The major function is follows:

* Measuring resolution better than 0.2mm

* Data window Size: 50x80(256x25) would be possible)

* Sample rate: Max. 15MHz

* Multipurpose: profile, energy spread, emittance measurement.

* High anti-interference:

When the beam intensity is so weak that the beam profile can not be observed on TV monitor, therefore it may be clearly seen on the graphic display after eliminating background noise from the image data taken repeatedly.

In order to record the 2ns beam intensity data which is important data for accelerator operator ,high speed sample hold circuits are being developed and we intend to use it instead of 7104 oscilloscope.

B.Timing system

PLS 60MeV LINAC timing system is a small system with three triggers to electron gun, travel wave tube, and modulator. In general, we refer to the LINAC timing system of KEK because the same kind system had been running for 5 years in the Beijing Election Positron Collider(BEPC) without trouble.

C.Beam monitors

According the physical requirements and our running experience in the BEPC, three categories of monitor are adopted.

The short pulse current monitor consists of a ceramic solid resistor in the shape of a disk, magnese-zinc ferrite aluminum case with BNC connector. Its features are:

* Measuring min. limit: 0.2 ma without amplifier

* monitor sensitive: 3mv/ma

* Frequency response: >1.5 GHz

The fluorescent target typed AF955 has been mounted in the profile monitor and can be movable by console computer. The beam loss monitor is not necessary for the short distance of 60 MeV LINAC, but it could be a prototype as reference for 2 GeV LINAC beam loss system.

IV. SOFTWARE SYSTEM

A.System software

The main control software on the Intel's 310 is the real time multitasking control software which is based on the BITBUS network. According to the LINAC physical requirements, it can carry out the control to each local station and make data processing. It owns its multitask scheduler. When the scheduler receives the command from the console, the related application tasks will be activated at any time. The control system uses fully operating functions, and the tasks will be put into operation in order of their priority level. The real-time data base is built in; it always holds refreshed data (over 300 signals) of whole control system. The data adjustment and command sending task is running forever after the control software is set up. It acquires the datum from each local station via BITBUS and updates the DB continuously in rate of 2-3 times/second.

B.Application software

In normal times, the control software in the local station is continuously acquiring the datum and monitoring from/to the accelerator's equipments. The major application software include as follows:

* Modulator/Klystron package [4] such as control and monitoring to modulator, drive power control, waveform digitalized control

* Magnet power control and monitoring

* Beam FWHM calculation and emittance processing etc.

The Human-Machine interactive software have been designed for those physicist and specialist who are not familiar with system software. It easy to operate and configure various control system. Please refer to "Human-Machine interface software Package" in this conference proceeding.

V. CONCLUSION

During the configuration of I&C system of 60 MeV LINAC, some technology, experiments and equipments such as beam monitors, MKSU, and timing are transmitted from KEK and SLAC. We believe that international collaboration has speeded up the progress of PLS 60 MeV LINAC.

Instrumentation and control system of PLS 60 MeV is designed for PLS's preinjector. So far, its commissioning with the whole machine will be in November, 1991.

It is a compact and complete control system for PLS 60MeV LINAC.

VI. ACKNOWLEDGMENT

We should sincerely appreciate Prof. W.Namkung, Prof. I.S.Ko, Dr M.H.Cho and I&C group of PLS for their direct support and friendly collaboration.

We should appreciate to Prof. K. Nakahara, I.Abe, A.Enomoto, T.Urano and Prof. Y. Kimura of KEK for their valuable experience and sincere assistance during the system configuration. Also we thank many friends at SLAC for their successful experience about Klystron control.

VII. REFERENCES

[1] K.Nakahara, I.Abe, R.P.Bissonnette, A.Enomoto, Y. Otake, T.Urano and J.Tanaka, "Control System for the Photo Factory 2.5GeV Electron LINAC", Nucl. Instr. and Meth. A251 (1986) 327.

[2] R.K.Jobe, M.J.Browne and K.P.Slattery, "Hardware Upgrade for Klystrons in the SLC", SLAC-PUB-3666 poster paper presented at the 1985 Particle Accelerator Conference

[3] D.K.Liu, K.Nakahara, I.Abe, T.Urano, A.Enomoto and J.Tanaka. "Image Processing System for Electron LINAC Beam Diagnosis", Contributed to the 1986 Linear Accelerator Conference, 2-6 June 1986, SLAC, U.S.A.

[4] R.K.Jobe, K.Thompson, and N.Phinney, "Klystron control software in the SLC", SLAC-PUB-3667

Multi-Microprocessor Control of the Main Ring Magnet Power Supply of the 12 GeV KEK Proton Synchrotron

T. Sueno, K. Mikawa, M. Toda, T.Toyama and H. Sato
National Laboratory for High Energy Physics

and S.Matsumoto,
Physics Division, Dokyo University School of Medicine

Abstract

A general description of the computer control system of the KEK 12 GeV PS main ring magnet power supply is given, including its peripheral devices. The system consists of the main HIDIC-V90/25 CPU and of the input and output controllers HISEC-04M. The main CPU, supervised by UNIX, provides the man-machine interfacing and implements the repetitive control algorithm to correct for any magnet current deviation from reference. Two sub-CPU's are linked by a LAN and supported by a real time multi-task monitor. The output process controller distributes the control patterns to 16-bit DAC's, at 1.67 ms clock period in synchronism with the 3-phase ac line systems. The input controller logs the magnet current and voltage, via 16-bit ADC's at the same clock rate.

1.INTRODUCTION

The main ring magnet power supply consists of 10 twelve-pulse thyristor rectifiers with dc filters, of 2 reactive power compensators [1] with tuned ac harmonic filters [2] and of an analog and digital hybrid control system [3]. A schematic diagram of the power supply is given in Fig. 1. Fig.2 shows the principle layout of the hybrid control system. Eight rectifiers feed the bending magnets and the other two excite the horizontally and vertically focusing quadrupole magnets. Fine adjustment of the current at injection and of the ratio between currents of the bending and the quadupole magnets is required to tune the acceleration. The current of the quadrupole magnets must be tracked separately from the current of the bending magnets for precise Q-tuning and optimum beam acceleration.

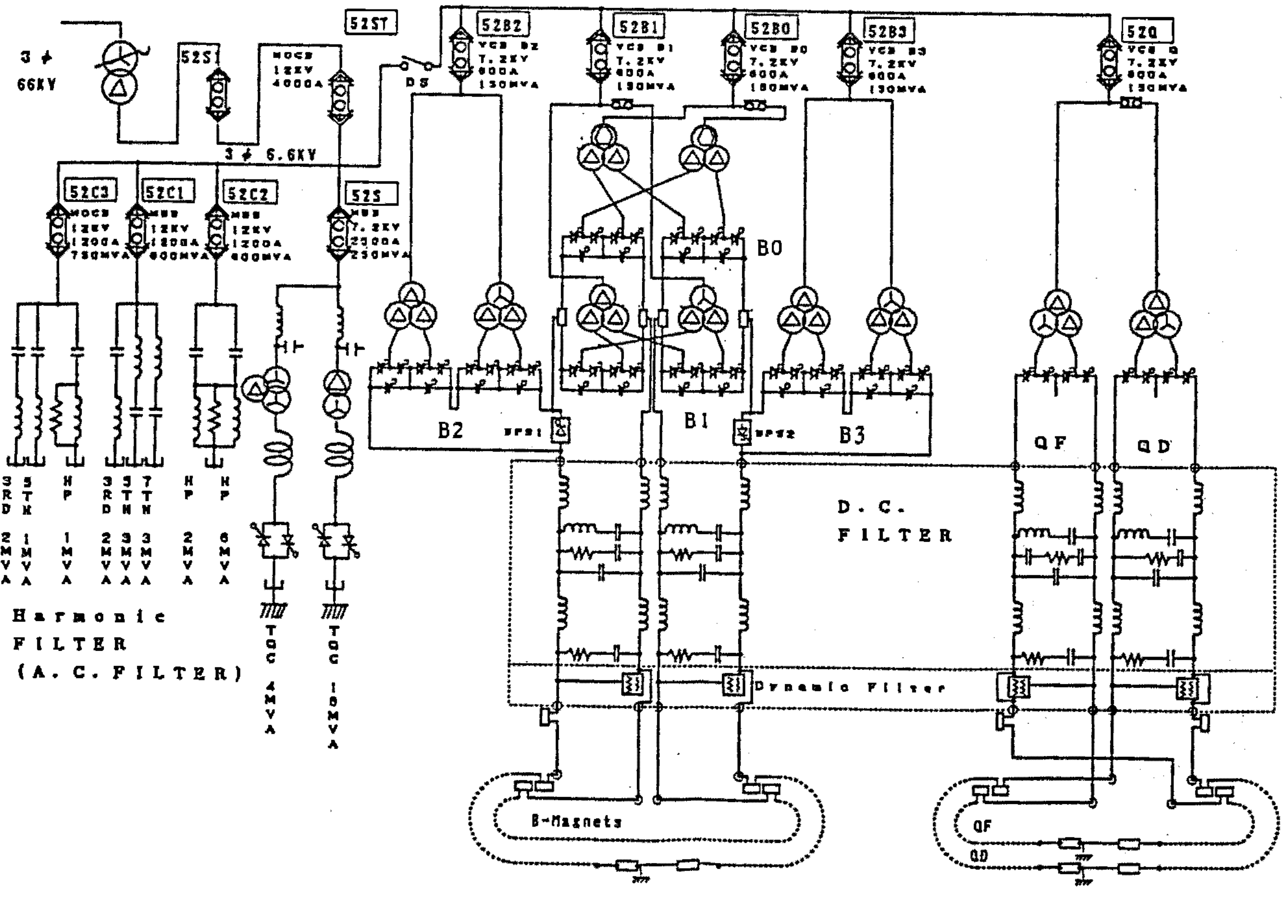


Fig.1 Schematic Diagram of the KEK 12 GeV PS Main Ring Power Supply.

The desired magnet excitation currents are obtained by controlling the output voltage of the thyristor power converters through the SCR gate firing pulse. The rectifier voltage reference patterns are implemented by the common action of two negative feedback loops, i.e. a low-gain automatic voltage regulator (AVR) and a high gain automatic current regulator (ACR). These patterns are elaborated by the control computer and fed to the regulation through a DAC, synchronized at 600 Hz on the zero crossing of the two 3-phase ac line systems. While providing the voltage reference patterns the computer implements a repetitive control algorithm on the base of measured deviations of the magnet current from reference. The digital system is in charge of the fast feedforward pattern control and of the repetitive control via the ACR.

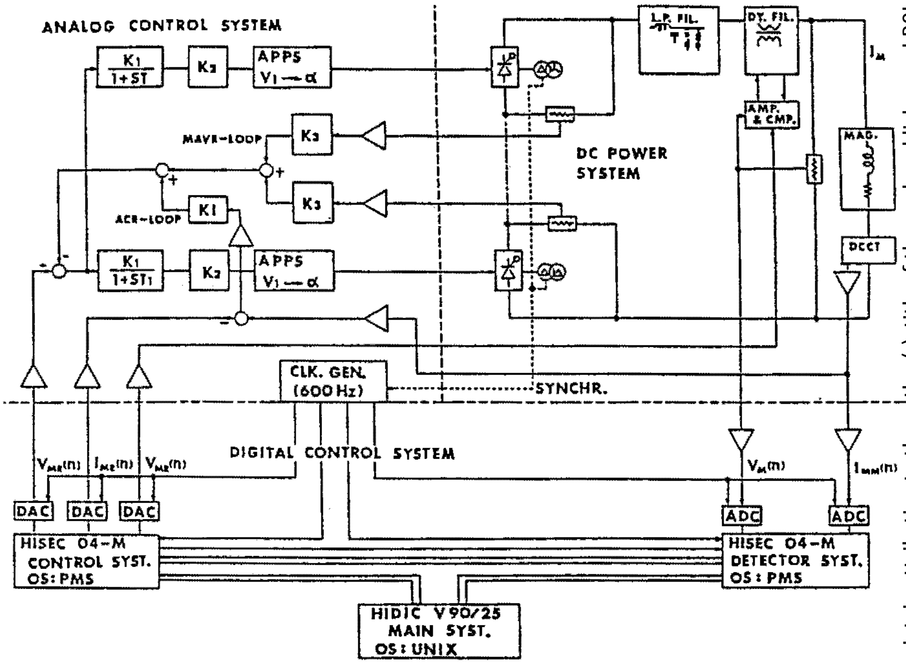


Fig.2 Schematic Diagram of the Hybrid Control System.

2. MULTI-MICROPROCESSOR CONTROL

2.1 General Layout

The multi-microprocessor control system has been introduced in 1985 [4]. Initially the main CPU was a V90/5 (8MHz without cash-memory); this was then upgraded to a V90/25 (16MHz with cash-memory) in order to improve the speed of the main system and of the communication loop. The main parts of the digital system are based on a 16-bit-microprocessor and on LSI components, connected to an industrial standard bus and to standard peripherals, supported by an universal operating system. Consequently a high level language and powerful utilities facilitate development and maintenance of flexible software for the pattern control system which consists of the main CPU HIDIC-V90/25 and of the input and output controllers HISEC-04M. The three distributed systems have no hierarchical software but are rather independent even at assembler level because of the difference between the CPU families and in particular of different addressing for memory access. The main components are LSI of the MC-68020 CPU family. The direct digital control system, as main part of the controller, consists of I-8086 and home-made LSI modules. Fig.3 shows a layout of the multi computer system.

2.2 HIDIC-V90/25 system

The system, equipped with memory management unit and 16MB DRAM on the internal bus, has a floating processor. Two local area network (LAN 1 and 2) loops provide the interconnection between main system and standard peripherals, i.e. 5-inch 80 MB hard disc and 8-inch 1 MB floppy disc drive, two CRT terminal stations, a typewriter and a printer. These resources are supervised by the UNIX compatible main OS. One of the network-loops, LAN 1, exclusively devoted to input or output network communication, transfers patterns or logged data between the main and the input or output controllers. The fact that this communication is rather slow, due to time sharing operation on the same system bus, limits the speed of response for fine adjustments.

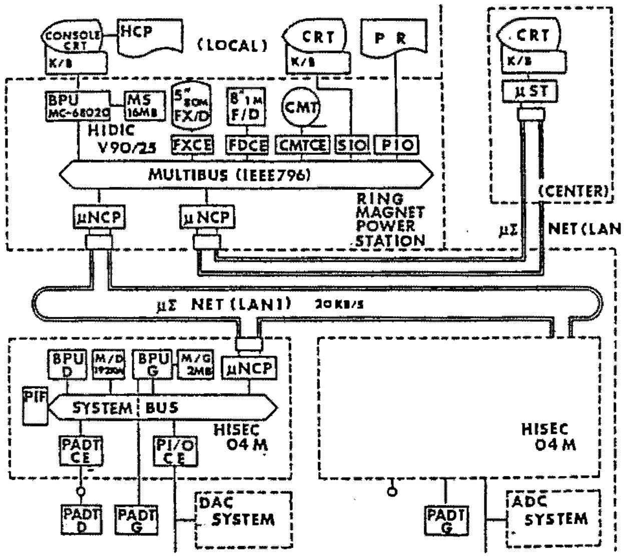


Fig.3 Layout of the Control Computer Netwok System.

The local terminal is supported by a serial I/O full duplex link. The remote terminal, located in the Center Control Room (CCR) of the 12 GeV PS, is linked by an optical cable to cope with the 350 m distance between CCR and power station, where the main system is installed.

The application programs are written in the system language C and FORTRAN 77 [5]. The parameter files of the programs displayed on a CRT terminal can be communicated to the central control system. Powerful UNIX utilities are used not only for program development but also for maintenance of the application programs, controlling and monitoring the whole system by file management, screen editor and shell command.

Pattern generation can be done while the power supply is running by using the main CPU and storing the new pattern in its memory. Therefore the operating pattern can be changed without interrupting power supply operation. Fine adjustment of the injection current and of the tracking ratio between bending magnet and quadrupole current is done from either terminal in the CCR or the local power supply control room. The main tasks are control operation, e.g. start-stop and status monitoring, calculating the correction patterns of the repetitive control and fine adjustment of pattern data. As far as additional supporting tasks concerns, the system works on pattern generation, processing of pattern and operational data, control program development and background processing. The main operator commands are shown in Table 1.

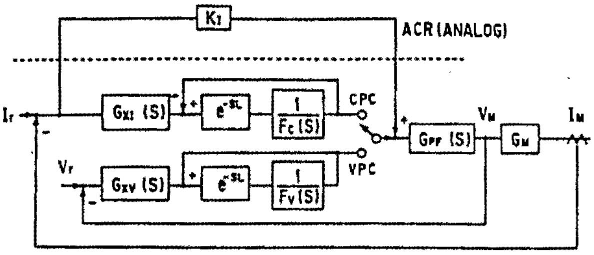
Table 1.

**** MR-PS OPERATOR COMMANDS****

f1 [pattern No.]	: run
f2	: stop
f3 pattern No.	: pattern exchange (with repetitive control)
f4	: IQ tracking adjust
f5	: B inj. adjust
f7	: pattern generation
f8	: PS status display
f9	: pattern No. display
f10 [pattern No.]	: pattern save
f11 pattern No.	: pattern remove
f33 pattern No.	: pattern exchange (without repetitive control)
menu	: command menu display
menu2	: command menu next page display
pc20	: repetitive control start
cpcstop	: periodic control stop

Footnote: [default pattern number] can be neglected

Generated pattern files are used for fine adjustment of injection level of the bending magnet current (without tune shift) by additional smoothing corrections calculated in the same pattern generation algorithm. Tracking offset calculations are done similarly to injection current corrections. Concerning the B2 and B3 rectifiers, the reference voltage pattern is subdivided and distributed to each of the 12 pulse thyristor converter groups in order to obtain the desired magnet voltage with minimum reactive power generation [6]. In both fine adjustment cases a step variation is smoothed out by applying optimum polynomials in a fixed interval. The main and the controller systems are linked in a LAN with an effective transfer rate of 20 kB/s. Typical response time, on fine adjustment of tracking or of injection current for beam tuning, was 20 to 50 s for the V90/5, even after the control programs have been optimized by fixed point calculation, but becomes less than 10 s in case of the V90/25. Table 2 shows as an example display of injection current adjustment by function f5 (see Table 1).



GX(n) : Transfer function for compensation.
F(s) : Transversal finite impulse response of low pass filter.
L=mT : Dead time of m times period T, m : integer.

Fig.4 Block Diagram of the Repetitive Control.

Table 2.

```
## MR-PS injection tuning ( on line ) ##
page - 1/1                  proton/q05066

     Ib    injection        : 198.88
     Iqf   injection        : 116.73
     Iqd   injection        : 116.29

                        Binj. [G] 1450.0

UPPER LIMIT > INITIAL > LOWER LIMIT
     Injection  [G]
     1598.9 > 1449.0 > 1285.5
```

Repetitive magnet current control has been performed to suppress the deviations from a given current reference pattern and repetitive voltage control has been used to reduce the parasitic voltage ripple [7]. The principle is based on the control method applied to a repetitive reference input [8]. The excitation current in the respective magnets is to be controlled according to periodical patterns. The frequency response of the correcting transfer function has been confined in a lower frequency region, about 15 Hz or less, to track a periodic input and assure the stability of the power system. Fig. 4 shows an algorithm of the repetitive control of current and voltage. In Fig. 5 the convergence of the repetitive control, from the initial pattern to the corrected one, for an ACR deviation of the bending magnet current pattern is shown over eight correction cycles. (Timing pulses, P1,P2,P3 and P4 are injection start, acceleration start, flat top start and flat top end, respectively.)

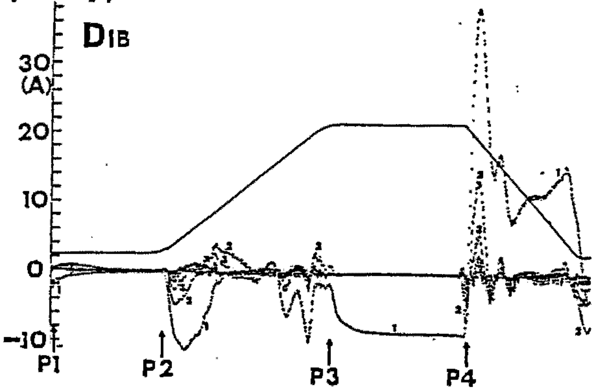


Fig.5 Converging of Current Deviation by Repetitive Control.

2.3 Controller System

The HISEC-04M/D and G controllers, belonging to different families i-8086 and HD-68000, are subdivided into two sub-system working on the same system-bus. The first one works as direct digital control and feeds operational patterns through 16-bit DAC's at 600 Hz clock. The other one acts as a data processing system and a support to the direct digital control; it logs the magnet current from the DCCT and the group voltage through a sets of 16-bit ADC's working at the same clock. It communicates through LAN 1 with the V90/25 and reads or writes data or messages on the memory of the HISEC-04M/D.

The output controller, supervised by the process monitoring system, executes application programs such as start-stop as well as process timing and sampling synchronization. The routine output processing function distributes 15 data patterns in memory through parralell I/O to the 16-bit DAC's.

Eleven sets of DAC's serve the bending magnet power supply: eight of them give the voltage reference patterns to the thyristor converter groups, two are for the ripple detectors of the dynamic filters and one gives the current reference for the analog ACR. Each focusing and defocusing magnet power supply has three sets of DAC's. Two serve as voltage and current reference to the analog loops and one is used for the dynamic filter. The system ouputs the pattern data to the seventeen sets of DAC at every 1.67 ms clock period and the data conversion is synchronized to the zero-crossing of the six phase ac power line. Control signals of by-pass thyristors and gate pulse suppress signals are distributed by the system.

The input controller is dual with respect to the output controller as far as hardware and system software concern, except the digital input and output. Its main task is to collect the data from the ADC's through paralell I/O and to accumulate and save them at every control clock.

Simultaneously the system reads data from six sets of 16-bit-ADC's through a Sample/Hold amplifier at the same clock as the DAC system. Three sets serve for the DCCT current signals and the others for the dc voltages applied to the B, Qf and Qd magnets. The clock is sychronized on ac voltage zero-cross pulses but has a constant delay of 100 microsec corresponding to about twice the conversion time. The DCCT current data serve in the repetitive control loop for calculation of corrections to the voltage references.

3. CONCLUSION

The hybrid control scheme of an analog and a digital system and the multi-microcomputer control system HIDIC-V90/25 and twin HISEC-04M have been implemented in the main ring magnet power supply of the 12 GeV PS. The HIDIC-V90/25 and HISEC-04M system perform fast feedforward pattern control at 600 Hz clock and slow but high gain feedback control of current pattern for steady deviations according to the repetitive control method. The analog system works as real time feedback control loop of the voltage and current reference pattern fed by the digital system.

The repetitive current control is not yet used routinely, despite its effectiveness, due to the cumulative effect of small errors produced by the intrinsic ripple of the present DCCT. It is used for initial pattern correction and fine adjustment of the injection current and of the tracking ratio between bending and quadrupole magnet currents. It is easy to change the operating patterns without stopping the power supply. During the extension of the flat top duration [9], memory and hard disc capacity has been increased and the application software modified. When operation is performed with a long magnet current flat top, the control clock is synchronized at 300Hz to save memory space. At present the multi-microcomputer control system allows to perform stable operation of the PS and to achieve effective utilization of the slow extracted beam spill.

Acknowledgements

The authors would like to thank Prof. Michio Nakano (Tokyo Institute of Technology) for the many technical discussions concerning the control system design . They are indebted to Prof. M. Kihara for his advice and encouragement during this work. They also thank many colleagues of PS group for their effective cooperation.

References:

[1] S.Matsumoto, H.Baba, H.Sato, T.Sueno and K.Mikawa, "Improved Control System of the Thyristor Flicker Suppressor for the KEK 12GeV PS," IEEE Transactions on Nuclear Science, NS30(1983)2932

[2] T.Shintomi, M.Masuda and S.Matsumoto,"Harmonic Current AC Filters at a Large Accelerator," Particle Acc. 8(1978)87

[3] S.Matsumoto, T.Sueno, K.Mikawa and N.Kumagai, "Control of the Main Ring Magnet Power Supply for the 12 GeV PS," Proceedings of the 6th Symp. on Acc. and Tech., 1987, Tokyo, Japan, pp.158.

[4] S.Matsumoto, T.Sueno, K.MIkawa, M.Toda and H.Baba,"New Multi-Microprocessor Control System of the Main Ring Magnet Power Supply for the KEK 12 GeV PS," Proceedings of the 1987 IEEE Particle Accelerator Conference, 1987, Washington D.C., pp.689.

[5] T.Sueno, M.Toda, S.Matsumoto and K.Mikawa, "Software Systems of the Main Ring Magnet Power Supply for the 12 GeV PS," Proceedings of the 6th Symp. on Acc. and Tech., 1987, Tokyo, Japan, pp.155.

[6] H.Sato, T.Sueno, T.Toyama, M.Mikawa, T.Toda, S.Matsumoto and M.Nakano, "Performance of the Main Ring Magnet Power Supply of the KEK 12 GeV Proton Synchrotron", Proc.of the 1991 IEEE Nuclear Science Symposium, 1991, Santa Fe, to be published.

[7] T.Sueno, S.Matsumoto, K.Mikawa, M.Toda, H.Sato, N.Kumagai and H.Baba,"Repetitive Voltage Control of the Main Ring Magnet Power Supply for the KEK 12 GeV PS," Proceedings of the 14th Int. Conf. on High Energy Accelerators, 1989, Tsukuba, Japan, Particle Accelerator. 29(1990)133.

[8] T.Inoue, M. Nakano, T.Kubo, S.Matsumoto and H.Baba,"High Accuracy Control of a Proton Synchrotron Magnet Power Supply," Contl.Sci. & Tech. for Prog. of Soc. (IFAC 82) Vol.6, 3137, Pergamon Press(1982)

[9] H.Sato, T.Sueno, T.Toyama, M.Mikawa, T.Toda and S.Matsumoto, "Upgrade of the Main Ring Magnet Power Supply for the KEK 12 GeV Proton Synchrotron", Proc.of the 1991 IEEE Particle Accelerator Conference, 1991, San Francisco, to be published.

VME COMPUTER MONITORING SYSTEM OF KEK-PS FAST PULSED MAGNET CURRENTS AND BEAM INTENSITIES

T.KAWAKUBO, A.AKIYAMA, T.ISHIDA*, E.KADOKURA
National Laboratory for High Energy Physics, 1-1, Oho, Tsukuba-shi, Ibaraki-ken, 305, Japan
*Mitsubishi Electric Company, 1-7-4, Iwamoto-cho, Chiyoda-ku, Tokyo-to, 101, Japan

Abstract

For beam transfer from the KEK-PS Linac to the Booster synchrotron ring and from the Booster to the Main ring, many pulse magnets have been installed. It is very important for the machine operation to monitor the firing time, rising time and peak value of the pulsed magnet currents. It is also very important for magnet tuning to obtain good injection efficiency of the Booster and the Main ring, and to observe the last circulating bunched beam in the Booster as well as the first circulating in the Main. These magnet currents and beam intensity signals are digitized by a digital oscilloscope with signal multiplexers, and then shown on a graphic display screen of the console via a VME computer.

1. INTRODUCTION

There are many pulsed magnets and beam monitors which concern beam injection and extraction of the KEK-PS-Booster as well as beam injection of the Main ring. In order to tune the machines and to search trouble points, it is very important to display these signals using proper trigger timing. Because we must select a proper signal and trigger among many connectors and choose a proper time scale, voltage range and trigger level, only a few trained crew members had been able to observe the expected signals within a short time. By using signal multiplexers, a digital oscilloscope with GPIB and a VME computer system, however, we can now observe the expected signals without any great effort using a touch panel of a console desk in the PS-control room.

2. PURPOSE OF THIS SYSTEM AND REQUIRED SIGNALS

A. Observing pulsed magnet current

In order to observe the operating conditions of the pulsed magnets, the following magnet currents should be observed with proper time scale:

- For Booster Injection:
  - four Bump magnets in series
- For Booster Extraction:
  - Bump(#1,#2)
  - Septum(#1,#2)
  - Kicker(#1~#4)
- For Main Injection:
  - Septum(#1,#2)
  - Kicker(#1~#5)

B. Checking the magnets' firing timing

For beam transport from the Booster to the Main ring, just after firing the Booster extraction septums and carrying out time-matching of an RF bucket of the Booster ring with one of the Main ring, Booster extraction bumps are fired; after about 20μsec, four kickers are fired at the same time. After a transfer time from the Booster to the Main, firing of five Main injection kickers follows. In order to check these timing, it is convenient to display the concerning magnet currents and a bunched beam intensity with a "mountain view"*.

---

*When you observe rapid changing figures as a kicker current and a fast beam intensity in the control room, the figure deterioration through a long co-axial cable becomes problem. We have re-shaped the deteriorated figure to the original by the "equalizer" made by Dr.S.Ninomiya. We would like to acknowledge him for his offering of his instrument.

- For Booster extraction:
  - a mountain view of currents of four kicker magnets and a pulsed beam measured by a wall current monitor (see Fig.1)
- For Main injection:
  - a mountain view of currents of five kicker magnets and a pulsed beam measured by a wall current monitor
- For all pulse magnet fire timing:
  - a mountain view of currents of septums, bumps and a kicker (see Fig.2)

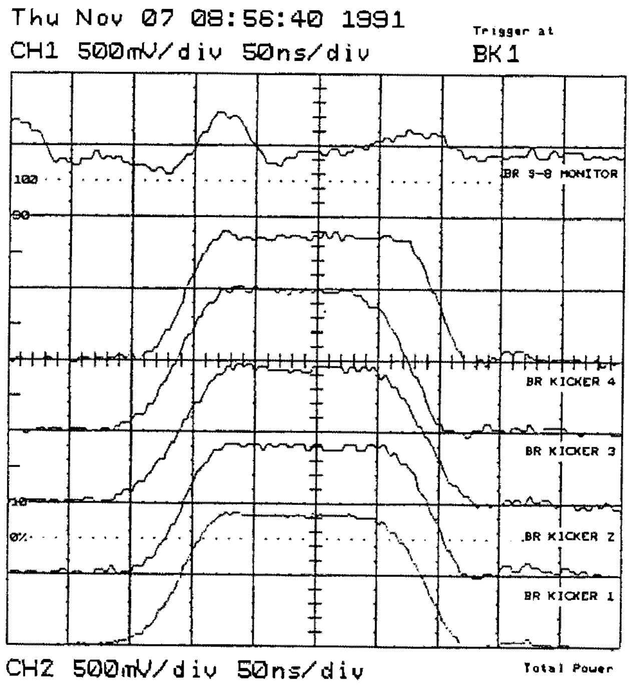


Figure 1. (from top figure to bottom)
A mountain view of currents of a pulsed beam measured by a wall current monitor, currents of four kicker magnet and averaged those four kicker currents

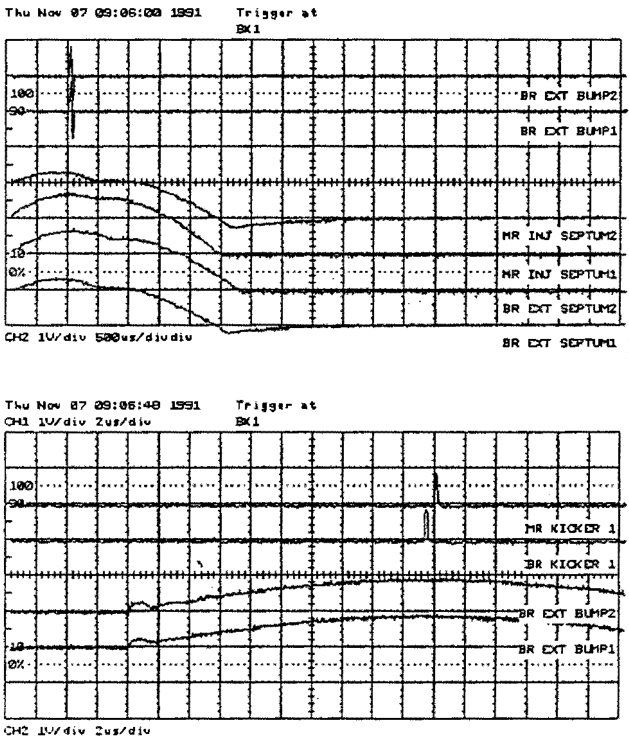


Figure 2. A mountain view of all kinds of pulsed magnets in PS-BT (from top figure to bottom)
Booster Extraction Bump (#2, #1)
Main ring Injection Septum (#2, #1)
Booster Extraction Septum (#2, #1)
Main ring Injection Kicker (#1)
Booster Extraction Kicker (#1)
Booster Extraction Bump (#2, #1)

C. Tuning machine

After tuning positions and emittance figures of a beam at the transport line, final tuning should be carried by observing the injection efficiency of the Booster and the Main. By dividing the Booster circulating beam particle number at injection by the Linac beam particle number, which is calculated by integrating the Linac beam current with time duration, the injection efficiency of the Booster is obtained. And by dividing the circulating particle number at the Main injection by that at Booster extraction, the injection efficiency of the Main is obtained. The fire timing of the kicker magnet should also be adjusted by observing the height of the Booster bunched beam at extraction and of the Main at injection. The trigger used to observe these monitors can be selected among nine successive Booster extraction beams which inject to the Main ring in the Main injection porch:

- For Booster injection efficiency:
  - Linac beam intensity and Booster particle number measured by a slow intensity monitor (see Fig.3)
- For Main injection efficiency:
  - Booster particle number measured by a slow intensity monitor and the Main particle number
- For Booster kicker firing timing:
  - fast intensity monitor at Booster extraction
- For Main kicker firing timing:
  - fast intensity monitor at Main injection

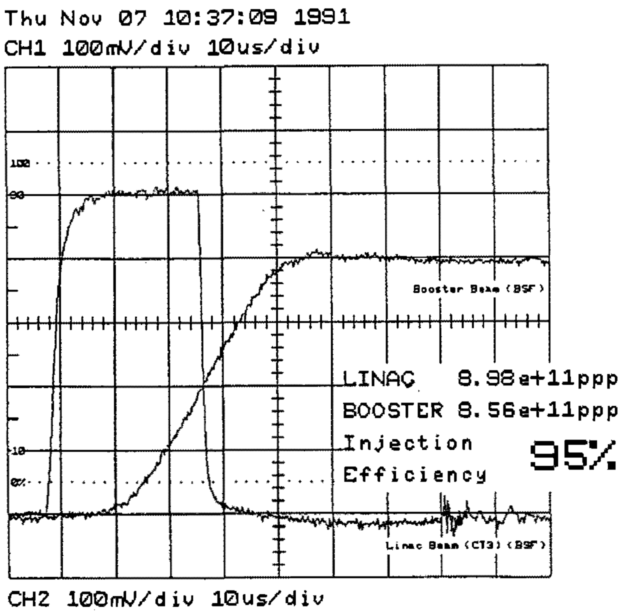


Figure 3. Linac beam intensity, Booster particle number at injection and Injection efficiency

D. Searching for trouble points of the bump and kicker systems

The thyratron used in a bump and kicker power supply has a lifetime when the turn-on timing becomes delayed. When one of the two Booster extraction bump magnets happens to show such a deterioration, the betatron amplitude arising from the bump firing increases. Therefore, two bump currents and ΔR monitor signal in the Booster are needed to check the problem.

Every kicker has a separate delayed trigger circuit, which has low reliability, and sometimes becomes out of order. When one of the kicker currents begins to be delayed, we can distinguish which causes the trouble (thyratron or delayed module) by changing the following trigger:

- For Booster extraction bump trouble (signal):
  - Booster extraction bump currents and ΔR monitor signal
- For kicker trouble (trigger):
  - origin of the trigger to fire all kickers
    - * for Booster extraction kicker
    - * for Main injection kicker
  - output of delayed trigger module after branching from the origin trigger
    - * Booster kicker(#1 ~ #4)
    - * Main kicker(#1 ~ #5)

3. BLOCK DIAGRAM OF THIS SYSTEM

In order to obtain the injection efficiency precisely, the Linac and Booster intensity, or the Booster and Main intensity, should be evaluated at the same time. Therefore, the Linac and Main intensity are connected to different inputs of an oscilloscope from that of the Booster intensity. The figures changing rapidly as a kicker current are connected to the input via the "equalizer" mentioned in the footnote. All of these input figures are displayed by using a proper trigger, as shown in Fig.4.

4. FUTURE IMPROVEMENT

We are using an oscilloscope with a sampling rate of 250MS/sec; the memory number is 1K words. This number is too small to display the synchrotron oscillation by taking the envelope of the height of the bunched beam train (because of "areasing" of digital oscilloscope).

We will purchase an oscilloscope with greater memory, so that we can display not only the synchrotron oscillation, but also a "mountain view" of a bunched beam train at the Main ring injection with an interval of a quarter of the synchrotron period.

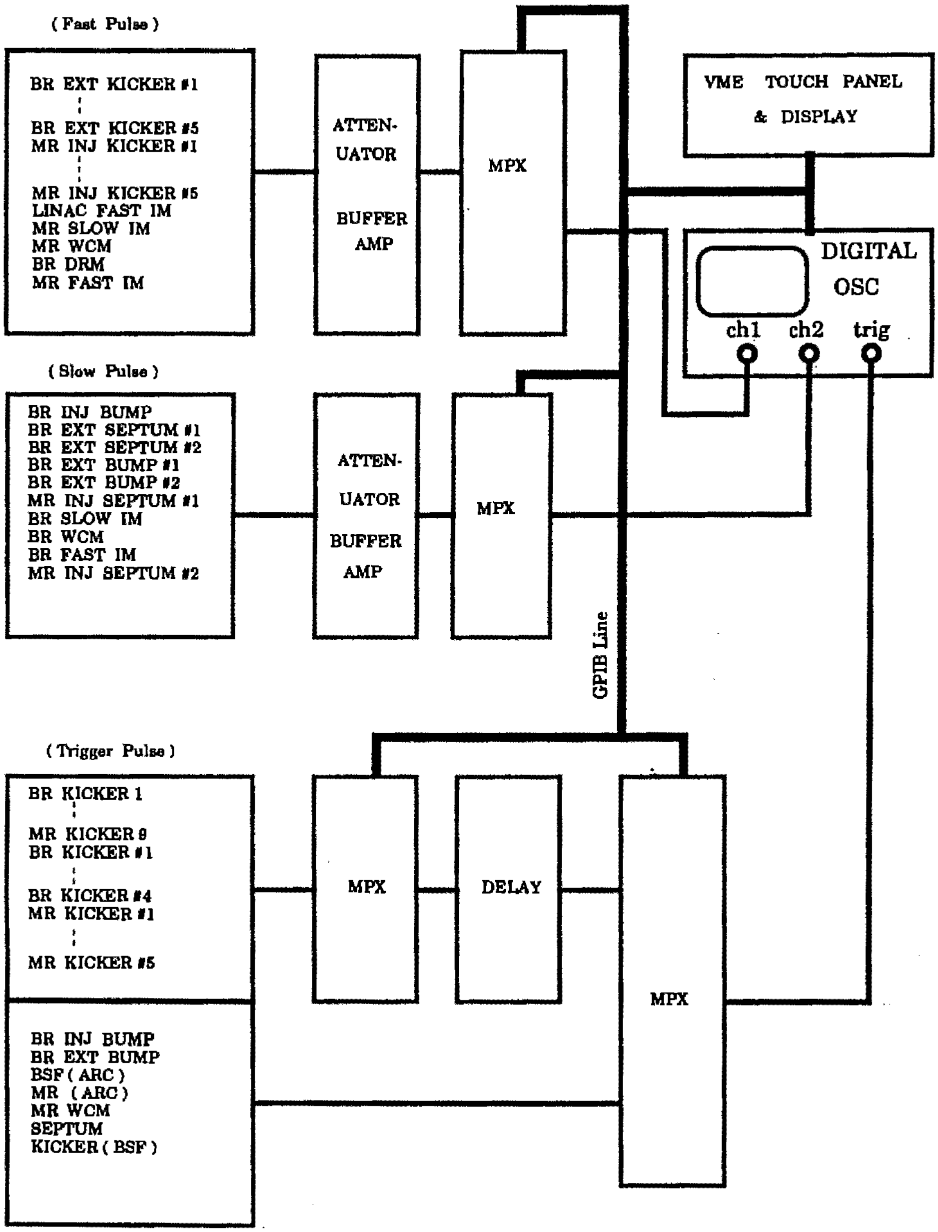


Figure 4. Block diagram for observing system of pulse currents
(BR:booster ring, MR:main ring, IM:intensity monitor, WCM:wall current monitor, DRM:delta R monitor, BSF:booster facilities, MPX:multiplexer)

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S04SRS09

Magnet Power Supply and Beam Line Control

for a Secondary Beam Line K6

Y.Suzuki, M.Takasaki, M.Minakawa, H.Ishii,
Y.Kato, M.Ieiri, K.H.Tanaka, H.Noumi,
and Y.Yamanoi
KEK National Laboratory for High Energy Physics
1-1 Oho, Tsukuba, Ibaraki 305, Japan

Abstract

K6 is a secondary separated-beam line with momentum range up to 2.0 GeV/c in the north experimental hall at the KEK 12 GeV Proton Synchrotron (KEK-PS). On the construction, newly developed magnet power supplies (MPSs), in each of them a microprocessor is embedded, are introduced. The features of the MPS are as follows:

1, The MPS is connected to an upper-level beam line controller (BLC) by GPIB highway for exchanging simple messages.

2, All the operations of the MPS are supervised by the microprocessor, which has its individual parameters and fault messages. It reduces the load of the upper-level controller.

3, The MPS has functions to inspect itself and to report the result. It saves much time and labor of maintenance.

INTRODUCTION

On the KEK-PS site, there are two experimental halls for high energy physics experiments. The one is the East experimental hall (E-hall), that has been servicing since 1977. The other is the North experimental hall (N-hall) built in 1990, where the construction of the new beam line K6 construction is under going.

In N-hall, the beam lines were designed for high-intensity proton beams against high radiation field. The R&D for the beam line components has made during the last few years. The design of the magnets and vacuum system were reported in [1]. On the other hand, magnet power supply (MPS) and control system have also been developing [2].

On the construction of the K6 beam line, newly developed MPSs were introduced. Each MPS has a microprocessor, ADC(analog to digital converter), DACs (digital to analog converter), relay I/O(input and output), and GPIB(IEEE-488) interface. The digital processing unit i.e. magnet power supply controller (PSC) is incorporated into the MPS to have functions; ON, OFF, reset of interlocks circuit, polarity switch, current/ voltage control mode, current setting with appropriate speed, and checking the health of the MPS without help of upper-level beam line controller. These functions are invoked by a simple message, for example; current setting message; "A 1234.0" means to set output-current to 1234.0 ampere. This message is sent to MPS from BLC through a single coaxial cable; GPIB highway.

These design concepts were reported several years ago, [3], [4]. Though effective, those design concepts have been applied to few devices on the accelerator field up to present.

We introduced this design concept in to beam line control system, and developed MPS. Now we are saving cost, time, and trouble.

This paper report this MPS's PSC and beam line control for K6. The details on soft program and hardware will be reported elsewhere.[5]

POWER SUPPLY CONTROLLER (PSC)

Hardware

PSC consists of five boards (STD: IEEE-961):

(1) CPU board: Z-80, GPIB communication interface.

(2) DAC board: 16-bit DAC, for reference voltage.

(3) ADC board: 16-bit ADC, for monitoring DCCT (output current).

(4) ADC board: 12-bit ADC, 16-channel multiplexer.

This board is used for monitoring the following values:

1, Reference voltage (16-bit DAC).
2, MPS's DC output voltage
3, MPS's output voltage for monitoring Wave Form.
4, MPS's AC input current .
5, Input voltage of firing module.
6, Seven points on low-voltage power supplies.

(5) Relay I/O board: for control and monitor.

The control points are:

1, ON/OFF
2, Reset of interlock logic.
3, Polarity switch.
4, Regulation mode (voltage or current).
5, Remote or Local

Input points for monitoring status are:

1, Control power ON/OFF.
2, Remote/ Local.
3, Main switch, ON/OFF.
4, Polarity +/-.

5, Magnet #1 Ready/ Trouble.
6, Magnet #2 Ready/ Trouble.
7, Fault; DC over current.
8, Fault; DC current leak,
9, Fault; MPS over heat #1.
10, Fault; MPS over heat #2.
11, Fault; cooling fan
12, Fault; MPS's door open.
13, Fault; magnet #1; over temperature.
14, Fault; magnet #1; cooling water flow trouble.

Software

All the PSC's soft programs are written in assembler language. They mainly consist of :

(1), GPIB communication program.
(2), MPS operation program.
(3), Watching program.
(4), Fault or error message list for diagnosis of MPS.

Messages [1] BLC--> PSC

The messages sent to PSC in MPS from BCL are the followings:

1), "ST$"
requests PSC to send status message of MPS. This message includes the following:

ON/ OFF,
REMOTE/ LOCAL,
READY/ NOT READY,
+/ - , : polarity
CC/ CV, : regulation mode
Reference value,
DAC value,
DCCT value,
Voltage value,
AC input current value.

2), "ST1$" , or "ST3$".
These messages request PSC to send MPS's status as follows;

DCCT value, or absolute output current [A].
Voltage value, or absolute output voltage [V].
+/ - , : polarity
CC/ CV, : regulation mode
Ac input current value [A].

3), "ST2$"
requests to send

Absolute output current [A],
Absolute output voltage [V].

4) "FL"
requests to send fault messages to help diagnosis of the MPS. Its message is text-format message.

5), "AC"
requests to send AC power source input current [A].

6), "OW"
requests to send wave form data of MPS's output voltage; 255 words binary data with EOI, its sampling period is about 25 millisecond. Figure 1 shows an example of the wave form.

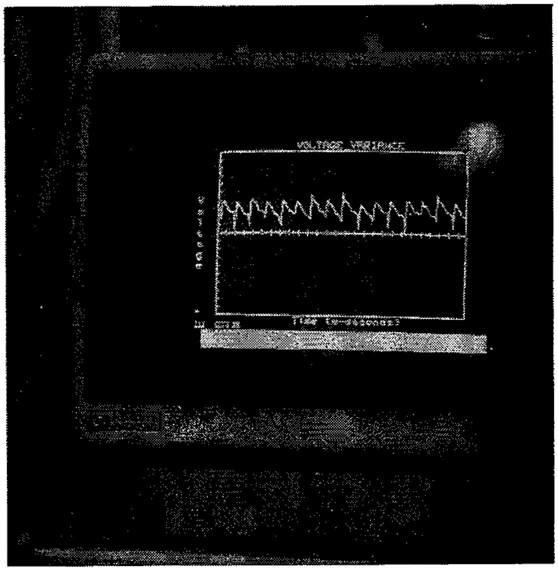


Figure 1: Wave form of MPS's output voltage

7), "ID"
requests to send MPS's identifying message;

It includes MPS's rating, factory name fabricated the MPS, etc.

8), "Y0"
requests to send DCCT output voltage for monitoring.

9), "X0","X1","X2", , , "X15"
request to send the value of ADC(with 16 channel multiplexer) for monitoring.

10), "A xxx.xx"
requests MPS to set output current xxx.xx [A] smoothly.

11), "V xxx.xx"
requests MPS to set output voltage xxx.xx [V] smoothly.

12), "D x" : (-10000<= x <= +10000)
requests PSC to set xxxx data to DAC smoothly.

13), "T1", "T2", "T3"
are setting speeds of output current or voltage.

T1: fast.
T2: middle.
T3: slow.

14), "L x"
sets the limit value of output current for watching.

15), "CC"
sets MPSs to constant current regulation mode.

16), "CV"
sets MPS to constant voltage regulation mode.

17), "CK".
It invokes check program to check MPS. When MPS is OFF state, the check program checks Interlock's fault signal, low voltage power supplies, DAC output voltage, and function of main switch and polarity switch. When MPS is running, it initializes check list and fault flag, then check program starts.

18), DC, SDC : GPIB command.
initialize PSC and MPS. Main SW is off, and all data is clear.

Message [2] PSC--> BLC

The messages sent to BLC form PSC have described already in the above section without SRQ.
The SRQ is a signal of PSC to request BLC. One status byte is sent to BLC.
The bit assignment is listed below.

Bit 0, 0: MPS is OFF state.
1: MPS is ON.
Bit 1, 1: Local control mode.
Bit 2, 1: Fault on interlock.
Bit 3, 1: Fault on ON, OFF, polarity SW procedure.
Bit 4, 1: Fault on Values; output current, low voltages, or DAC: reference voltage.
Bit 5, 1: Message error.
Bit 6, 1: SRQ
Bit 7, 1: This bit is set after the check program runs, and indicates that the fault-list is available for read-out.

MPS operation

All the operation of MPS is done by PSC in MPS. When current setting message is received, the MPS's operation is as follows.

Step 1, Receive "A xxx.xx".
Step 2, Invoked MPS operation program.
Step 3, Reset MPS interlocks.
Step 4, Check Interlocks fault signal.
Step 5, Check MPS's low voltage power supplies, MPS output current, and DAC output voltage for reference.
Step 6, Set MPS to current or voltage regulation mode.
Step 7, Check polarity, and turn polarity switch.
Step 8, Main power switch ON, and check.
Step 9, Current setting and check trouble loop.
Step 10, Current setting end, and SRQ.
Step 11, Set limit values for watching program.
Step 12, Watching: check status (ON/OFF, interlock signal, remote or local), output current, reference DAC, and low voltage power supplies.

On this state, when MPS receives "A-xxx.xx" message, the following steps are done.
Step 13, Current setting starts to 0 [A].
Step 14, Main power switch OFF.
Step 15, Jump to the above step 2.
Final state is negative (-) xxx.xx [A].

On the step 15, when MPS receives "V+ xx.x" message, the following steps are done.
Step 16, Current setting starts -xxx.xx[A] to 0 [A].
Step 17, Main power switch OFF.
Step 18, Jump to above step 2.
Final state is positive(+) xx.x [V] on voltage

On the OFF state, when MPS receives "CK" message, the following steps are done.
Step 19, Reset interlocks, check fault signal.
Step 20, Check output voltages of low-voltage power supplies, DAC, and DCCT(output current).
Step 21, Turn polarity switch, check its status.
Step 22, Turn main switch ON, check its status.
Step 23 Check output voltages of low-voltage power supplies, DAC, and DCCT.
Step 24, Turn main switch OFF, check its status.
Step 25, Send SPQ, bit 7 added in order to indicate check-end.

BEAM LINE CONTROL

MPSs and control system for K6

The control system for K6 using new MPSs is shown in figure 2. All the MPSs are connected through GPIB highway to the BLC by a coaxial cable. A terminal display connected to the BLC is offered to a user group to operate the beam line. The BLC computer is dedicated for the beam line control and the MPS's maintenance work.

On construction stage of the beam line, this new MPS's function is effective for checking MPSs. All the function of MPS is performed through the microprocessor embedded in the MPS. Then commands or messages to the MPS is simple, those MPSs can be operated easily by direct command of personal computer with interpretive language (BASIC) too. The diagnostic information of the MPSs are able to get without checking program on the BLC, for its checking or test program is stored in each of MPSs.

On this configuration BLC programing becomes simple, and no checking program is running constantly on BLC. As the result, the GPIB communication line between MPSs and BLC becomes quiet, there is no problem with speed.

The equipments for K6 beam line control system are:
BLC: Personal computer with BASIC language HP-300
TRM: Terminal for user of K6 beam line.
EX: GPIB bus extender for long distance up to 1250m, 60k bytes/s. HP 37204A
MPSs: listed bellow.

Name	Address	KW	A	V
D1	1	260	2000	130
Q1	2	65	1300	50
Q2	3	105	1600	65
Q3	4	200	2000	100

Q4	5	160	2000	80
CM1	6	85	1300	65
CM2	7	85	1300	65
Sext	8	50	1000	50
Q5	9	65	1300	50
Q6	10	85	1300	65
Q7	11	105	1600	65
Q8	12	120	2000	60
D2	13	500	2500	200
Q9	14	120	2000	60
Q10	15	160	2000	80

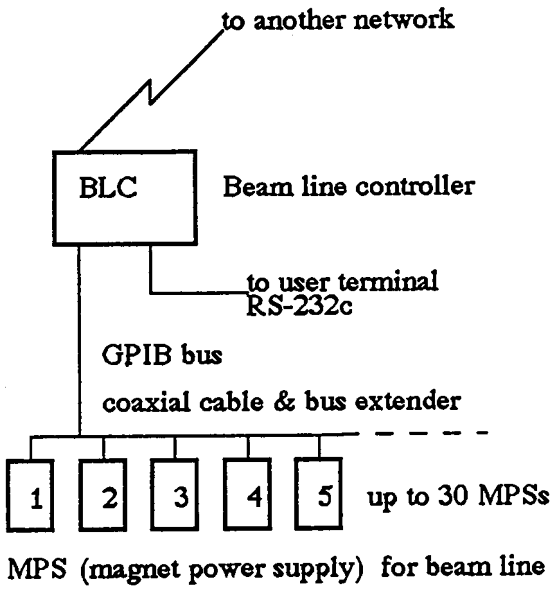


Figure 2. Configuration of K6 beam line control

The soft program for K6 beam line is written by BASIC language, which does not include the operation procedures or the fault messages of MPSs, therefore it becomes simple.

The operation procedure or the specifications or the fault messages of MPS are MPS's own. It is better that they are not separated from MPS's body, because they form MPS's character. If they were separated, in case of worst case of MPS exchanging, it needs more work, e.g. exchanging of procedure program and fault messages for individual MPS in BLC.

CONCLUSION

We have developed new MPSs, and introduced the MPSs in K6 beam line. Both the specifications and the source program of the PSC were offered at free to factory or workshop for MPS fabrication. By using this type of MPS, we rationalized the work on control wiring, check and maintenance work of MPSs, and BLC programing.

No longer on BLC programing the programer need to be concerned with the specifications or the fault messages or codes of MPSs.

ACKNOWLEDGEMENTS

We would like to thank professors K.Nakai, S. Kurokawa for encouragement and special aid.

REFERENCES

[1] K.H. Tanaka, etc all, The Beam-Handling Magnet System of the KEK-PS New Experimental Hall: MT-12, June 23-28, 1991, Leningrad, U.S.S.R.
[2] Y. Suzuki, M. Takasaki, Development of a Computer-controlled Magnet Power Supply for KEK PS Beam Lines: NIM, A293, pp. 253-257, 1990.
[3] M. Crowley-Milling, Control Problems in Very Large Accelerators: IEEE NS-32, No. 5, pp. 1874-1876, Oct 1985.
[4] J. H. Humphrey, The Isabelle Control System-design Concepts: Distributed Computer control systems, Proceedings of the IFCA Workshop, Oct 1979.
[5] Y.Suzuki, M. Takasaki, PSC hardware and soft program, KEK internal paper.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S04SRS10

SPECIFIC BEAM DELIVERY SYSTEM OF MEDICAL ACCELERATOR HIMAC

S.Minohara, T.Kohno, M.Sudou, M.Endo
T.Kanai, F.Soga and K.Kawachi

Division of Accelerator Research
National Institute of Radiological Sciences
Chiba , JAPAN

Abstract

A specific beam delivery system for radiation therapy in HIMAC is being designed. This report describes an outline of the beam delivery control system and its operation.

I. INTRODUCTION

HIMAC is a heavy ion accelerator facility designed for radiation therapy[1]. Beam delivery system of HIMAC is very specific and different from the ordinary facilities for experiments of general physics. The treatment control system for irradiation of patients is closely linked with operation of accelerator and beam transport. We report an overall idea of HIMAC beam delivery system and its operation for radiotherapy.

II. CLINICAL REQUIREMENTS

The clinical requirements for radiation therapy are described as follows.

1) At the end of irradiation, three dimensional dose distribution at the tumor volume in the patient must be achieved with an error of less than a few % compared to the precalculation of the dose distribution by the physician. Above all, overdose to the patient must be absolutely avoided.

The tumor of the patient as a target is set at the beam iso-center with an accuracy of less than 1mm. In case of the abdominal organ as the target, it is subject to move by breathing, and the margin of irradiated field should be considered in the treatment planning. Since the shape, volume and position of patient's target and the planned dose distribution are different for each patient, setting of many kinds of devices in the beam port varies at the time of each irradiation. The size of most treatment is satisfactory within a maximum field of 22cm in diameter which is based on clinical experiences at NIRS. On the other hand, small fields such as less than 1cm are often required. Hence, devices of beam port must be accurately adapted for wide range of field size.

2) Irradiation time per patient must be less than a few minutes. The reason is that the patient is immobilized on the couch by the shell or capsule, and immobilization of longer time gives much stress to the patient with illness. Now we are estimating that it takes about ten minutes to set a patient for positioning on the couch. Therefore, treatment time that a patient stays in the treatment room is about fifteen minutes. HIMAC has two synchrotron rings and three treatment rooms (Fig. 1). In the room B, horizontal and vertical beams can be utilized at the same time, and the room A and the room C have the vertical and the horizontal beam course respectively. Accordingly, two beams are delivered to four beam ports alternately. The course of each beam is changed at interval of about ten minutes, and the beam should be immediatly adjusted in compliance with medical requirement for each patient. To realize such a rapid change, all magnets along the beam transport lines are actuated by corresponding treatment schedule and the beam course can be changed by setting only one switching magnet in HEBT(high energy beam transport) line. For these reason, switching magnet must have accurate reproducible setting of field strength and high stability.

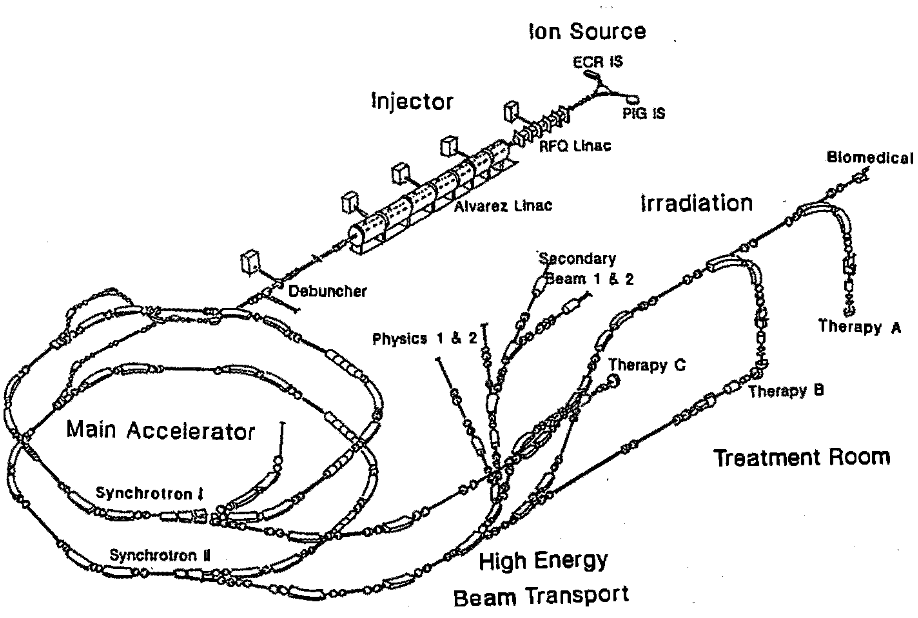


Fig. 1 A schematic view of accelerators and beam lines

3) HIMAC is expected hopefully to be utilized for more than fifty patients everyday for clinical treatments. In order to realize this, the treatment schedule and the flow of patients must be managed to run smoothly.

4) All systems in HIMAC are designed under the consideration of primary importance for keeping safety of patients and collaborating staffs. Especially, interlock system associated with the beam must be carefully designed from the view point of reliability.

III. TREATMENT SYSTEM

A schematic view of the treatment control network that we designed is shown in Fig. 2. The treatment system of HIMAC consists of the following components.

A. Treatment Planning System

In order to make the best use of heavy ion beam's characteristics, we are developing a three dimensional treatment planning system using a super graphics workstation (TITAN750) [2]. This planning system consists of:

1) defining target volume and critical organs by interactive contouring on Xray-CT, MRI and PET images,
2) determination of directions and shapes of irradiation fields using beam's eye view graphics,
3) calculation and display of three dimensional dose distributions,
4) designing collimators and compensators,
5) generating digitally reconstructed radiographs which are used for patient positioning.

In addition, the treatment supporting computer converts the planning data to the treatment control data, which consist of beam parameters, port data and treatment couch parameters.

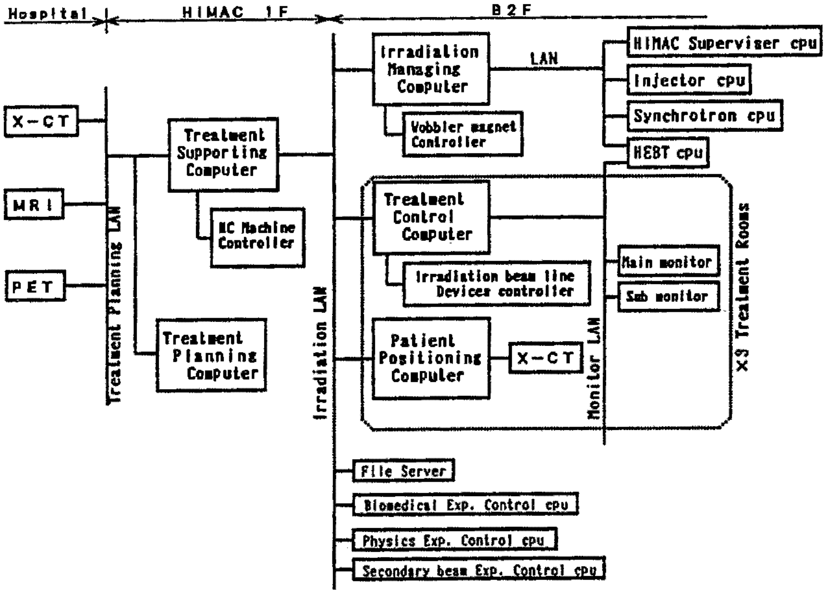


Fig. 2 HIMAC treatment computer network

B. Patient Positioning system

For patient positioning, we usually use the laser pointers, light localizer and digital Xray TVs that are incorporated in the beam port. The three dimensional coordinates of the target are estimated by coordinates of anatomical landmarks in the process of reconstruction by two projected Xray images perpendicular to each other[3]. Referencing the digitally reconstructed radiographs that are generated at the planning, the transfer of treatment couch is determined. The couch is to be operated by the treatment control computer(HP9000/380) linked to the patient positioning computer(HP9000/730) with image devices. Further, CT scanner can be used for the check of patient verification.

C. Irradiation system

Irradiation managing computer(HP9000/380) communicates with HIMAC central supervisor computer. It gives a requirement of the beam course, beam energy and ion species to HEBT controller via supervisor along the schedule. Concerning the responsibility of beam irradiation, we use the name "RIGHT" which means the initiative of opening the neutron shutter and the FCN(one of the Faraday cup monitors) shutter. The irradiation , that is the on/off of the beam, should be under the control of treatment side. Usually, irradiating at the treatment room starts after RIGHT is transfered to the treatment control.

Devices of irradiation beam port (Fig. 3) comprise a pair of wobbler magnets, a beam scatterer, a range shifter, a ridge filter, a multileaf collimator and monitors. They are controlled by the treatment control computer via interface unit.

To protect from accidental irradiation to patients and collaborating medical staffs, interlock systems are carefully built against such occasions as probable overdose, various kinds of troubles in each instrument, change of beam intensity and change of patient's condition including his unexpected movement. Further, quick stop and restart of irradiation are required repeatedly for medical use. In consideration of these things, operation of opening and closing the neutron shutter and the FCN shutter are handled either automatically or manually. They are synthesized in the form of global interlock system which is driven with threefold safety through hardware and software.

IV. OPERATION OF IRRADIATION

A layout of treatment room floor is shown in Fig. 4. Chief radiation therapy technologist(RTT) sitting by the irradiation managing computer can always look over the treatment hall and watch the ITVs coming from treatment rooms. In addition, the status of patients and beam lines are displayed on his console. So he manages the schedule of irradiations for smooth running.

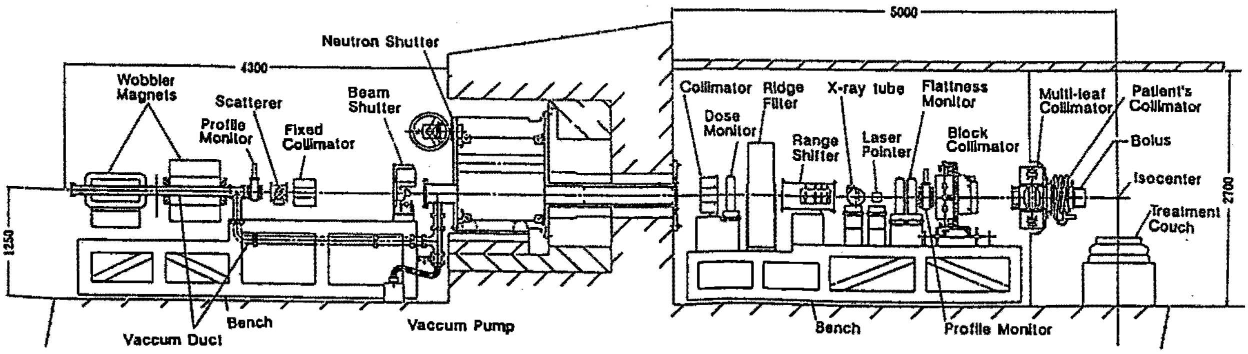


Fig. 3 A layout of beam port in the horizontal line

In ordinary clinical use, every morning, at first, RTT calibrates monitors in the same conditions of various devices just as the irradiation for each patient. Each parameter file which contains beam course, beam energy, ion species, setting parameters of magnets and so on is saved in the treatment control computer. This file name is checked before corresponding patient's irradiation, and the irradiation starts in the same condition at the time of calibration. The patient's irradiation starts after these.

Following the check of patient's ID-card, devices of beam port are automatically set on the starting status. While the RTT is setting the patient on the treatment couch, operators of HEBT adjust the beam up to the position before the neutron shutter, and keep waiting as the beam is stopped by FCN shutter. After the patient's setting, the RTT goes out from the treatment room and closes the shield door. The status of shield door is connected to global interlock system. And next, the RTT requests the RIGHT to HEBT. After the RIGHT is transferred to the treatment control, he opens the neutron shutter. It takes about ten seconds to open. Then RTT opens the FCN shutter which takes less than one second, and the irradiation starts. Besides global interlock system, whenever RTT wants to suspend irradiation depending on the patient's condition, he can shut the beam and restart quickly afterward. Once the dose counting of main-monitor reaches preset counts, kicker magnet kicks off the beam, and then FCN and neutron shutter are closed through the interlock system. A main-monitor system is always backed by sub-monitor system. Then RIGHT returns to HEBT with the data of request to next beam course. The data of irradiation records such as final counts of monitors, times of suspension, positioning images and status of port devices are preserved for each patient.

Now, we are designing and developing the software and hardware to control the beam for radiation therapy in HIMAC. The clinical trial will start in 1994.

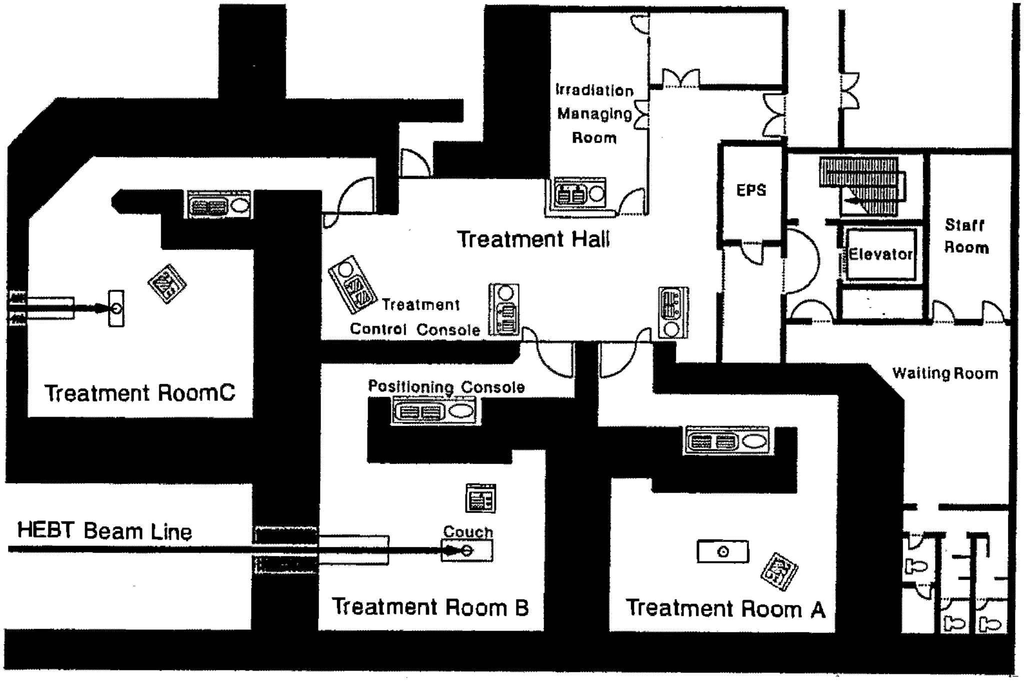


Fig. 4 A schematic view of treatment floor (HIMAC B2F)

References

1) K.Kawachi et al., Proc. 2nd Int. Sympo. Advanced Nuclear Energy Research, p.362 (1990)
2) M.Endo et al., Proc. of Int. Conf. on MBE, vol.29, p.317 (1991)
3) S.Minohara et al., Proc. of Int. Conf. on MBE, vol.29, p.859 (1991)

A CONTROL SYSTEM
FOR A FREE ELECTRON LASER EXPERIMENT

D. GIOVE

INFN MILAN -LASA, Via Flli Cervi 201 Segrate (Milan), Italy

Abstract

The general layout of a control and data acquisition system for a Free Electron Laser experiment will be discussed. Some general considerations about the requirements and the architecture of the whole system will be developed.

I. INTRODUCTION

The aim of the ELFA (Electron Laser Facility) experiment is to study the physics of a single pass FEL amplifier operating in the high gain Compton regime using a short electron pulse beam. The experimental purpose is the production of high peak power (0.3-1 GW) of microwave radiation, with a basic wavelength of λ_r=3 mm, and the possibility of tuning from λ = 1 cm to λ_r=0.1 mm. In order to achieve this goal an electron beam of very high current (400 A) in short pulses (6 cm) and with a maximum energy around 10 MeV will be injected into the wiggler midplane.

The accelerator consists of two sections: a photocathode injector providing a 3.5 MeV beam and a superconducting LEP II module to increase the energy up to 10 MeV. The wiggler will be a composite one, consisting of two coupled sections: the first part made with an hybrid structure (iron poles and permanent magnets) and a second part with an e.m. structure. A complete review of the project is given in [1] and a general layout of the experiment is showed in fig. 1.

The ELFA project has been funded by INFN and a lot of work has been done in order to define the conceptual design of the major components and to deeply investigate the FEL physics.

II. BASIC CONTROL PHILOSOPHY

A preliminary analysis of the characteristics required to the control system for ELFA pointed out the following items:

- ELFA needs both a control and a data acquisition system. Since ELFA is an experiment itself it is mandatory to have a complete data acquisition system for the measurements which have been planned to verify the basic ideas of the project. It is not possible to separate machine operations from physicist work. The two systems must be designed at the same time, sharing , as much as possible, the same philosophy and allowing an easy exchange of data.

- ELFA would take at least one year to "freeze" the characteristics of the major components. Nevertheless before of these period the control philosophy has to be fully developed and tested, in order to be an intrinsic feature of every component. A control system must play a central role in the whole design of a machine, to be really effective and to justify its budget requirements,. It is an old-fashion, money-wasting philosophy that one which consider the control system as just an "add-on" of the machine. In this way the control equipments just duplicate features already present and does not provide any improvement in performances. At the same careful attention has to be paid in order to evaluate the trend of development of computer technology. One has to balance today requirements and needs, with tomorrow availability and costs. This is more difficult to do since the different growth rates of the two basic components of a computer system : hardware and software.

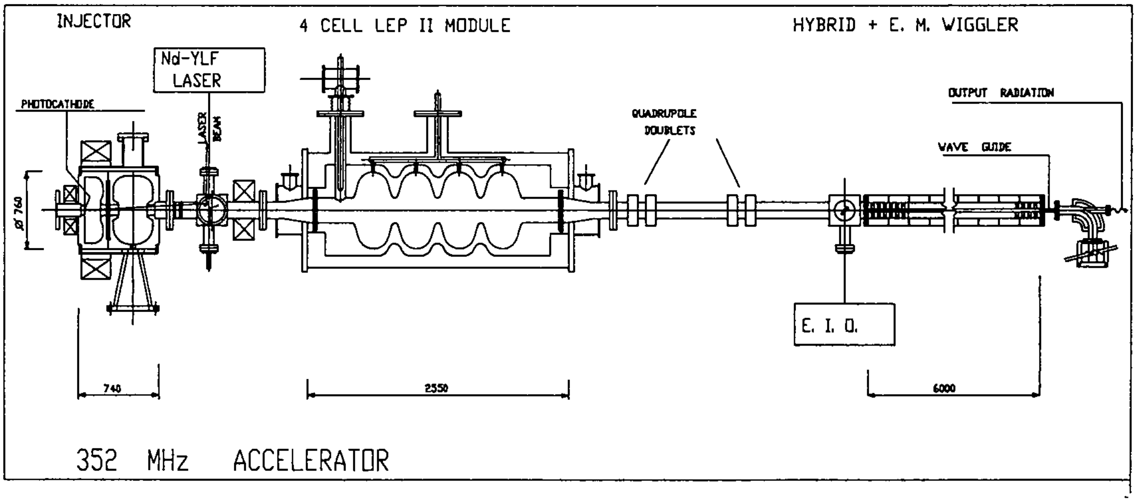


Fig.1 -General layout of the ELFA experiment

- ELFA would experience during its lifetime a lot of developments and the overall structure would follow this evolution. This statements outlines one of the most stringent requirements on every accelerator and experimental physics control system. Unlike the other basic subsystems of the machine, which will experience a limited amount of developments, the control system is required to be flexible and to satisfy a lot of requirements, that during the design phase have never been examined or discussed. This implies that a system that just meets today needs will not be able to work tomorrow. A lot of attractive self-contained systems are today available on the market, usually based on Personal Computers. These systems have an exciting first look but the experience gained working with them teaches that there is a not negligible risk to be tied from their internal structure.

- ELFA requires some general tools which will help to manage the whole project. Since the different subsystems would start their design and testing phase in a parallel fashion, it is very hard to take into account all the news coming from the different teams and to coordinate all the jobs. The control system in an experiment of this sort would be a valuable help in the analysis phase. Models definitions and simulation of the different views of the experiment would be possible using automatic tools. General rules and specific requirements would be cross referenced, pointing out inconsistencies or helping in optimizing performances.

Keeping into account all the points above discussed a conceptual design of the control and data acquisition systems has been made. The most relevant characteristics may be so summarized :

- the systems will be based on a fully distributed environment. Distributed systems represent an ideal answer to demands about flexibility and modularity and may be easily implemented using commercial standard products. The availability of components which follow international or DE-facto standards represents a great improvement in the way to think a control system. The attention may be moved from the home made development of basic components, which is a really expensive and time consuming activity, to the definition of general rules for the integration of commercial products in a multivendor, suitable environment.

- The systems must follow general rules both for hardware and software components. An extensive analysis of the general requirements coming from the different subsystems must be carried out in order to define the smallest number of different boards and software modules to deal with. This task is particularly difficult in small projects where there are a lot of single components.

- The user of the systems must be unaware of the details of them. The duty of the control group is to provide an high level environment for the programmer, being a physicist or a member of the technical staff, to let him to develop application programs, which require his scientific or technical skill, without have to deal with device or control architecture characteristics. All the operations on the equipments or on the experimental areas should be carried out using logical names related to the specific functions. No matter if they are on different control subsystems, or if they are moved from a controller to another one during system modifications. All the data must be available to the physicists when they need them in a simple and standard way.

- The development of the control system should be based on an extensive use of automatic tools. This would start in the design phase, with a special emphasis on software, and continue up to the installation tests. A lot of informations must be recorded and analyzed during the project evolution to guarantee an homogeneous environment. This will enforce the use of advanced programming techniques giving the possibility to the user to take full advantage of such an approach.

III. CONTROL SYSTEM ARCHITECTURE

Regardless of the real complexity of the experiment, modern distributed control architectures develops on three levels: the plant level,where we have the single equipment control; the process level, which is responsible for the operation of a set of functional related devices; the supervisor level with the operator interface. This general scheme has been adopted also for the general layout of the ELFA control and data acquisition systems (fig. 2). The main difference between the two systems is the absence of the plant level for the data acquisition system .

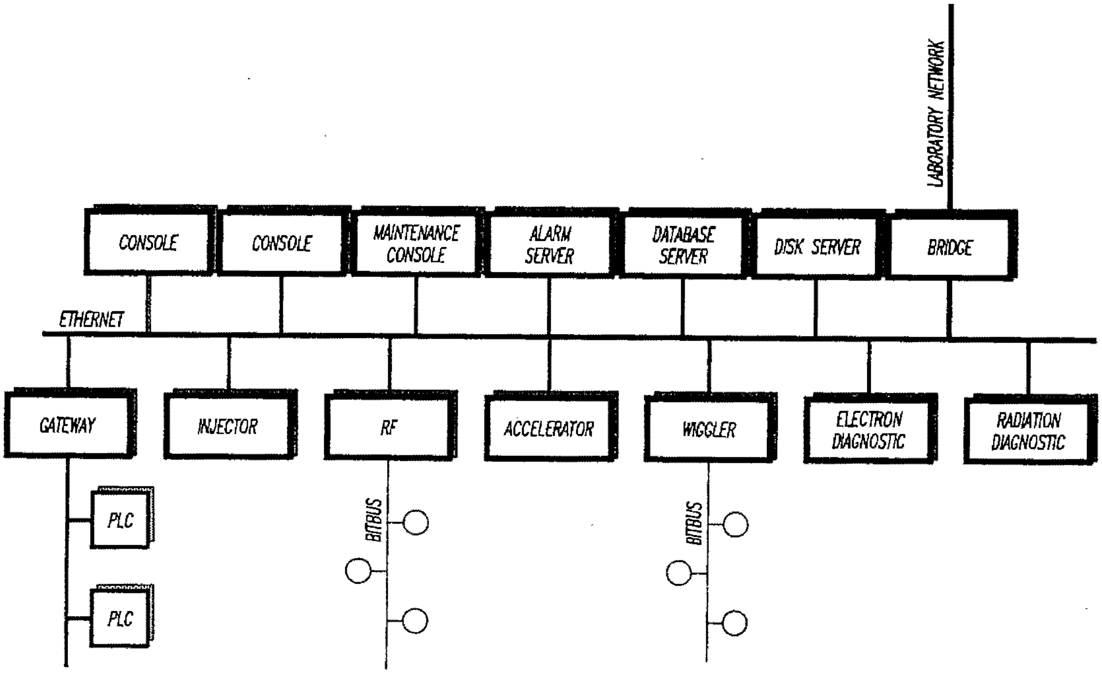


Fig. 2 - The hardware architecture of the control system

The process level has been designed as a set of Process Units (PU) connected on the main Ethernet network of the control and data acquisition systems. The PU are based on the VMEbus and the Motorola 680x0 microprocessor family. At this level the main tasks are to acquire and process data from the plant level, for the control system, or from detectors, for the data acquisition system and to process them according to operator commands or control codes specific of the particular PU. According to the specific tasks which should be performed a PU may follow a single processor or a multiprocessor scheme. The choice of VME as the internal bus for process level stations, among other buses similar from the technical point of view, has been due to the world-wide diffusion of this bus, specially in the research environment, which gives the possibility to work with a lot of boards and drivers already developed. A special consideration would be dedicated to the control of alarms conditions arising from equipments which will compromise their safety. To handle these situations we have planned to use industrial PLC connected to the main network by means of a gateway.

The plant level has been designed as a set of microcontroller based boards. Each board is housed in the equipment to which is devoted giving an enormous flexibility to the whole structure and providing a first processing of field data close as much as possible to the field itself. Boards related to the same functional part of the machine will be connected by means of a bus, implemented using an HDLC based protocol (Bitbus), to the related PU. This scheme reduces in a valuable way the pick up of noise in signal transport and helps in device hiding.

The operator level will be implemented using dedicated workstations. This choice is quite a standard one nowadays and the experience gained using these machines is really satisfactory [2].

IV. SOFTWARE CHARACTERISTICS

The overall structure of the control and data acquisition systems has been carefully analyzed from the point of view of software requirements. Two items seems to be the most important :
- the capability to configure in every moment of the experiment the whole software structure in order to reflect modifications in the hardware or in the requirements from the experiment
- the possibility to develop programs without having to deal with the distributed nature of the control architecture.

Control programs and data acquisition tasks (which are stored in a dedicated computer) will be configured to reference to logical names and tables which every CPU will load at the bootstrap or following a specified command. This structure allows an enormous flexibility during operation of the machine. Tables will be extracted from a central database which stores all the informations related to the project. The choice to have a database where the whole data of the machine are stored plays a central role in the design of the whole machine. We are dealing with a tool based on a CAD program which directly interface to the database : designing a piece of equipment each relevant information may be directly stored or retrieved for cross reference checks. A particular situation is that one when the software itself in under development : we may analyze it using SA-SD models and store the structure and objects we require for those particular tasks in the database. At the end of a module one may start programs which try to find semantic or conceptual errors. Moreover it is possible to extract a detailed and complete set of informations which constitute the specs for writing the software. In this way we have both a graphical look at the structure of the code and a documented version of the code itself.

To handle data and commands exchange on the network it has been decided to use as much as possible the RPC protocol as developed for the LEP control system. This choice let us to meet immediately the goal to hide the network structure to the user and to obtain a reliable task to task communication in such a way that a program running on a PU may call a procedure to be executed by another PU. RPC protocol provides the possibility to communicate between PU and the operator worksites. It is not possible, as by now, to access directly an I/O point at the plant level using our scheme. This would be possible writing special drivers on the PU where the microcontroller boards are connected.

The operating system we have chosen for the process level is OS-9, while at the plant level an Intel kernel, iRMX 51, provides support both for multitasking operations and for message exchanges.

The operator workstations will use a Motif interface and will run a dedicated program where the operator will be able to build its own interface or use a predefined view of subsections of the experiment. The choice to use workstations as operator access point to the control system takes into account performance and budget considerations. The only possible alternative to the use of a workstation would be to use Personal Computers for low level jobs and maintenance purposes. As by now, we have experienced that a PC would use the facilities of the RPC protocol if it uses Xenix as the operating system. As a tool during equipment design and test, PCs would be used in a stand alone fashion with DOS as operating system. National Instruments boards and LabWindow software will be the products of choice for these purposes.

Acknowledgements
I wish to thank the whole control group of ELETTRA-Trieste for a lot of interesting discussions on trends in control architectures and for their help during the first tests of the devices described in this paper.

REFERENCES

[1] R. Bonifacio et Al. "The ELFA Project" in EPAC Proceedings, Nice, France, June 1990, p. 340
[2] D. Giove et Al. "The Operator View of the Superconducting Cyclotron at LNS Catania" paper at this Conference

Control System for JAERI Free Electron Laser

Masayoshi Sugimoto
Japan Atomic Energy Research Institute,
Tokai-mura, Naka-gun, Ibaraki-ken, 319-11 Japan

Abstract

A control system comprising of the personal computers network and the CAMAC stations for the JAERI Free Electron Laser is designed and is in the development stage. It controls the equipment and analyzes the electron and optical beam experiments. The concept and the prototype of the control system are described.

I. INTRODUCTION

The Free Electron Laser (FEL) facility, SCARLET (SuperConducting Accelerator for Research of Light Emission at Tokai), is now under construction at JAERI[1–3]. It is a first step of the FEL program and the aim is the R&D of the superconducting accelerator (SCA) based FEL system in 10–50 μm range. The SCA is employed due to the suitability for the cw operation in the second phase of the project. The layout is shown in figure 1 and the main characteristics are described in table 1.

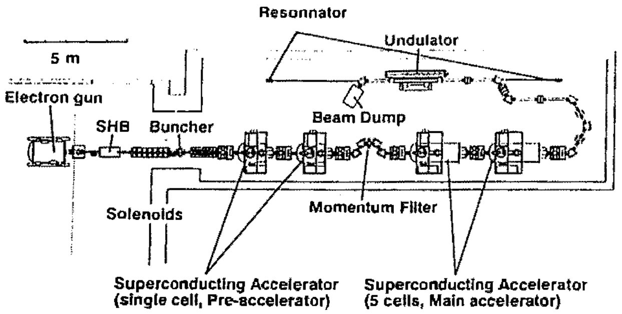


Figure 1. Plan view of the accelerator room of the JAERI free electron laser facility.

The accelerator itself is a small size with less than 20 m length, however, it would be expanded to 60 m in the second phase. In the design of the control system of the JAERI FEL, which will be also used in the next phase, the requirements were posed as:

1. Flexibility for evolving the system,
2. Reliability of hardware and software,
3. User interface for operator console,
4. Integrity of control and simulation,
5. Distributed control for fast response.

Table 1
Main characteristics of JAERI FEL
(first phase)

ITEM	SPECIFICATION
Electron Energy	14 – 23 MeV
Energy Spread	< 0.2 %
Peak Current	> 10 A
Pulse Width	40 ps
Repetition	10.4 MHz
Undulator Pitch	≈ 3 cm
Laser Wavelength	10 – 50 μm
Laser Peak Power	1 MW

II. HARDWARE ARCHITECTURE

In this project, the flexibility has the highest priority to accommodate the frequent change and upgrade of the hardware devices in the development stage and at the next phase. The devices in the facility are divided into three subgroups: (1) the injector section (electron gun, sub-harmonic buncher, buncher, and injection beam transport line), (2) the accelerator section (superconducting cavities, rf power supply, refrigerators, momentum filter, achromatic bend line) and (3) the optics instruments (undulator, mirrors, optical detectors). They are well isolated by the locations and their functions.

Each subgroup is controlled by a local unit equipped with:

- a 32-bit personal computer (NEC PC 9801:cpu i80386 16/20MHz, 3–5 MB RAM, 14-in CRT) with minimal peripherals to control the local unit alone,
- a dedicated CAMAC crate system with parallel bus crate controller, which contains analog i/o, digital i/o and GPIB interface modules,
- an Ethernet interface.

The main console unit consists of two personal computers, one is used to control the tasks in the network and another is used to analyze and display the acquired data or on-line processed results in a 21-in CRT. These are connected by Ethernet and SCSI for accessing the common 300 MB hard drive and 600 MB magneto–optical drive. As the pointing devices, mice and a touch panel are used.

Fig. 2 shows the hardware configuration of the control system for the JAERI FEL facility schematically.

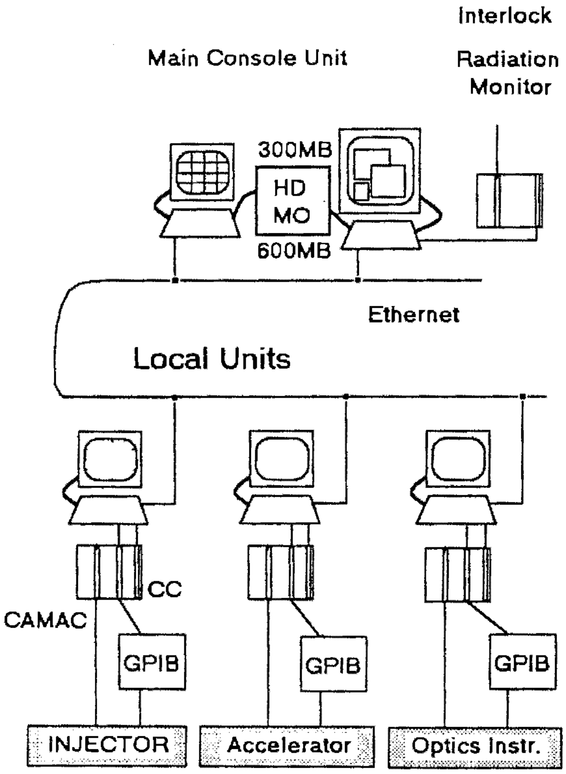


Figure 2. Schematic diagram of the hardware configuration.

The reliability of the hardware is important to maintain the laser oscillation, which is very sensitive to the electron beam quality. As the hardware considerations, we employ the modular approach in each layer: (a) the device driver level, which uses the standard interfaces, such as CAMAC and GPIB, and can be backup by other modules, (b) local unit computer level, which consists of the same units and it can be configured to share the task in the multiple computers, and (c) network level – if it is necessary, the network can be also doubled using an optical link of GPIB or serial communication at the expense of the data transfer speed.

The user interface is a severe problem when we employ the personal computer as a console and use graphics extensively, because display has lower resolution and the cpu ability is smaller compared with the workstations. To resolve the overload problem, the tasks to control devices directly are distributed over the network nodes (local units) and the console works only for organizing the messages among them.

The main console unit has an associated computer to display the auxiliary panels for supporting the control. It computes the beam characteristics using simulation codes, if required, and helps the operator to manage the knowledgebase on the machine operation. This causes to integrate the device control and simulation calculation in the FEL oscillation experiments.

The response speed is also a critical issue when the beam quality is fluctuated in a short time span. In the first phase of the project, the accelerator is operated in pulsed mode with 1 ms macro pulse width and 10 Hz repetition, so it is preferable to remedy the malfunctions found at the prior macro pulse in the 0.1 s pause. Basically, the feed back loops are inside of the device controllers, and it is designed that the CAMAC/GPIB data acquisition speed and the network speed are not critical.

III. SOFTWARE ARCHITECTURE

The system software consists of a multitask real–time kernel, drivers for the network, CAMAC and GPIB interfaces, and a graphical user interface. It has two modes of operations: simulation and on–line modes. In the development stage, the former mode is used to abstract the details and evaluate the system performance. In the actual operation, it is turned to the latter mode.

The flexibility and the reliability are enhanced by employing the object–oriented approach for software development. The software for FEL system control is configured in a hierarchy of three levels: *(1) process level, (2) hardware system level,* and *(3) hardware device level*, as shown in fig. 3.

As an instance, it is assumed that the task of process level control to transport in a portion of beam line is issued by an operator and it is requested to stabilize the condition of the electron beam (e.g. the position and the size). The process class has a sub class of hardware system level, one of whose instances is a double bend achromat in the current beam line. The hardware system comprises of the several sub classes of the hardware devices and one of the instances is a specific dipole magnet.

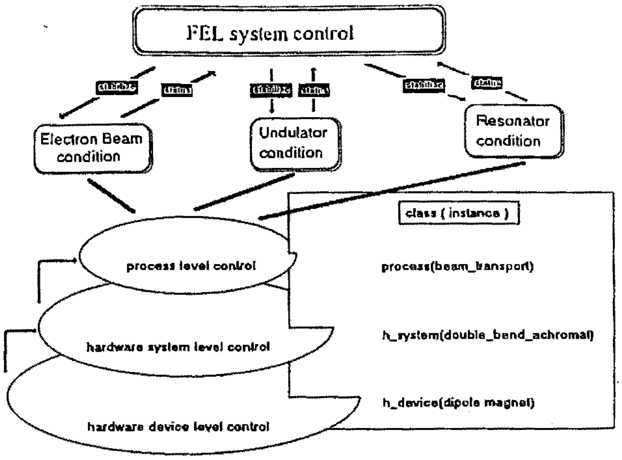


Figure 3. Software hierarchy of FEL system control.

To satisfy the request, the database of the magnet is searched and the corresponding control variable is found, then it is adjusted according to the rules sensitive to the objectives (beam position and size). There are many variables related to the hardware system so that they synchronize and communicate to each other. The final results are reported to the console as the status. And if required, the precise information about the history of the processing is recorded and reported the reasoning.

In the system database, or knowledge base after learning the rational rules of the operations, the classes of objects are represented by frames. Figure 4 shows an example of such a representation. It is described by the Prolog style. The "is_a" predicate represents an inheritance between classes, and "create"/"destruct" makes/deletes an instance. As the option, the multiple inheritance is attractive for categorizing the control items with the different aspects, such as the control method, speed, quality and quantity. The independent database is used when the informations are known to be orthogonal and the speed of the search is important.

As an example, the frame for bending magnet in a beam line is shown in figure 4. There are static and dynamic slots. And the dynamic slots are either the measured values or the derived values. The derived values can be controlled by a specified rule or formula. The slot can be created or deleted by using "assert" or "retract" predicates.

The integrity of the control and the simulation codes[4] can be attained by "beamtrace" or "matching" predicates, which produces an estimated beam condition or the optimized parameters during the control. These functions are combined to "stabilize" some characteristics. And "status" and "history" are reported as the resultants.

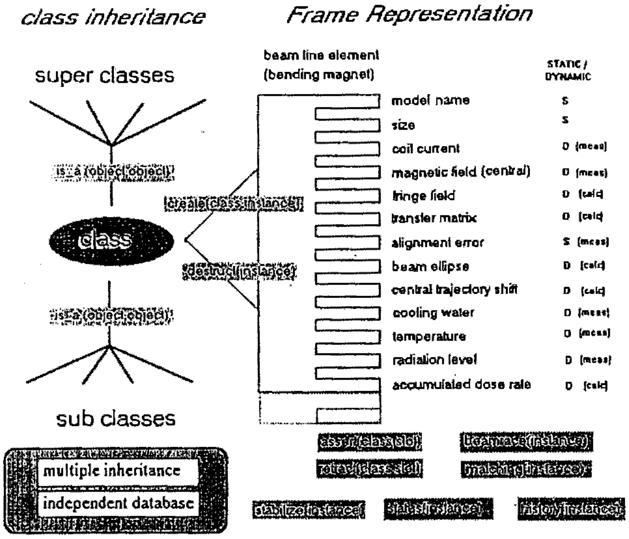


Figure 4. Representation of the system database in FEL system control.

The user interface for operator console is designed based on the graphics representation. Figure 5 shows an example of the panel for control. It contains all parameters of the electron gun, two buttons, three slide bars and four meters. The parameters can be controlled by using a mouse.

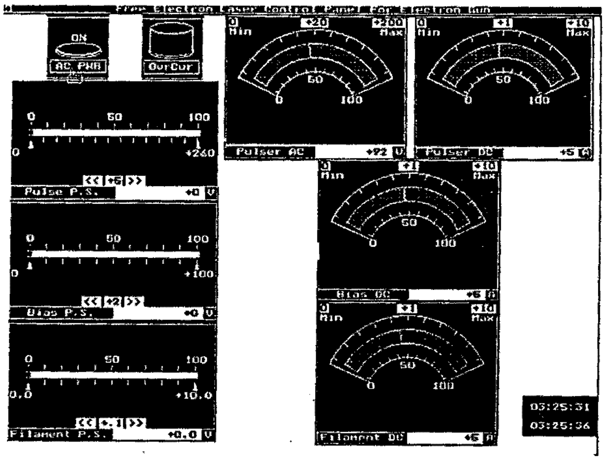


Figure 5. An example of the control panel.

IV. CONCLUSION

The most important feature of the JAERI FEL control system is flexibility for evolving in the future. So the hardware is selected in the frame of the standard interfaces, CAMAC, GPIB and Ethernet, and the software is organized using the object-oriented approach. The control process is combined with the simulation code to support the operator to know the theoretical background of the current status and acquire the experiences on the operation procedure. The construction is not yet completed, but the control system may be used by hiding the unimplemented hardware with the simulated (virtual) devices.

REFERENCES

[1] Y. Kawarasaki et al., "Linac for a Free Electron Laser Oscillator", Nucl. Inst. and Methods, A285, pp.338–342, 1989.
[2] M. Ohkubo et al., "Status of the JAERI FEL System", Nucl. Instr. and Methods, A296, pp.270–272, 1990.
[3] M. Sawamura et al., Proc. of 13th Free Electron Laser Conf. Santa Fe, Aug. 25–30, 1991 (to be published in Nucl. Instr. and Methods).
[4] M. Sugimoto and M. Takao, "Beam Transport Program for FEL Project", *ibid.*

Control Software for the ESO VLT

G.Raffi, European Southern Observatory (ESO)
8046 Garching, Germany

Abstract

The Very Large Telescope (VLT) project of ESO consists of an array of four optical telescopes of 8m diameter, to be installed at a new site in the Atacama desert in Chile starting in 1995.

The control software is completely distributed, being based on LANs interconnecting microprocessors and workstations, where several users and operators will be active at the same time.

Microprocessors are used in a variety of control functions, including the active control of the shape of the main mirror and compensation for atmospheric turbulence. Dedicated links and antennas are planned for direct communication and remote observation from various European centers.

The main concepts and novelties of the software design are explained.

I. THE VLT PROJECT

A *Main characteristics*

The VLT project has been initiated with the aim to provide European astronomers with a ground based telescope of larger size than those presently available [1].

The VLT concept consists of an array of four identical main telescopes, each having a thin monolithic mirror of 8m diameter. This gives an equivalent total size of 16m, when the four telescopes are used together, which shall be the largest size available on ground telescopes at the end of the 90s. The large size of the array will allow a very high angular resolution (the possibility to resolve details).

Each main telescope has an Alt-Azimuthal mount and is equipped with instruments at the two Nasmyth foci, Cassegrain focus and Coudé focus.

The four main telescopes can be used independently, or in several combined modes. One of the combined modes is the incoherent beam combination in a combined Coudé laboratory.

The other combined modes foresee coherent beam combination in an interferometric laboratory. In this case the use of two or more auxiliary telescopes of 1.8m diameter is also foreseen. These should be moveable on tracks. Optical path differences will be compensated with the use of delay lines and the optical beams are brought to interfere. Astronomical images can then be reconstructed starting from the interference patterns.

The use of a chopping secondary mirror has been proposed.

The VLT site is the Paranal mountain in northern Chile, in the Atacama desert at an altitude of 2700m.

Figure 1 shows the telescope layout foreseen for the top of mount Paranal. The first telescope is due to be installed at the end of 1995, with each following telescope being installed at time intervals of one year.

The whole VLT program including instrumentation will be concluded some years later. This gives a very long timespan for the installation of the VLT and of the control system in particular.

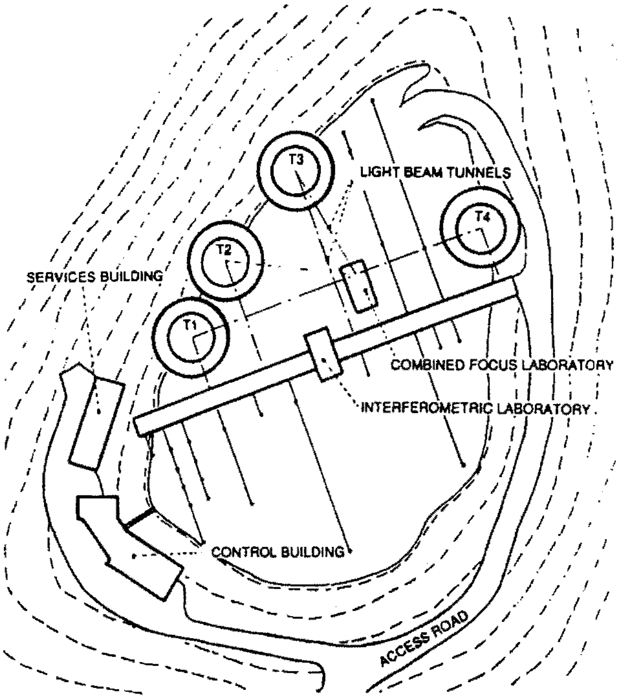


Figure 1: VLT observatory

B *Special aspects*

Special aspects of the VLT telescope with respect to other telescopes are:

- system distribution

  This is implied by the fact that the VLT is an array of four telescopes.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S05SRN03

- active optics

  This is a system of 150 supports for the main mirror, which allow it to change its shape. It is meant to compensate for deformations due to manufacturing errors, gravitational and thermal effects and also low frequency wind load effects on the main mirror.

  The image quality is analysed via a wavefront analyser, and a microprocessor provides the values to the support system.

  The active optics principle has been tested on the New Technology Telescope (3.5 meter diameter) now in regular operation at the ESO La Silla Observatory in Chile. It is thanks to this principle that not only excellent optical quality can be achieved, but also that substantial savings on the weight of the mirrors and on the size and costs of the whole telescope can be made [2].

- adaptive optics

  This system is introduced to compensate for the effects of atmospheric turbulence on the quality of images.

  It consists in applying to a deformable mirror a number of piezo-electric actuators, which compensate in real-time in a control loop for the optical degradation of images.

  This principle has been tested at the ESO 3.6m telescope, but not yet used on a regular basis as part of any instrument.

  The associated multichannel feedback loops require fast and powerful computation. It is acticipated that, for the VLT, human interaction can be eliminated, and procedures can be defined to allow routine use of this system [3].

C *Operational context*

The VLT operation has to cope with a complex environment produced by a multi-telescope, multi-instrument and multi-user context.

- Multi-telescope context

  The VLT software will support the following telescope configurations:

  - Stand-alone configuration

    Each telescope is used alone, allowing individual observing with an 8 m unit telescope.

  - Combined incoherent configuration

    This configuration includes the use of two, three or four main telescopes working either simultaneously on the same object with identical instruments, or via a combined instrument at the Combined Coudé focus.

  - Combined coherent configuration

    This is the configuration used for the VLT Interferometer (VLTI).

    It requires the use of at least two of the four VLT telescopes and of one or more of the auxiliary telescopes. Important aspects are the precise control of the position of the moveable auxiliary telescopes and of their associated delay lines.

- Multi-instrument context

  Normally up to three instruments will be mounted all the time on the main telescopes and additionally one has to consider the combined instrument and the VLT interferometer instrumentation. Therefore the VLT is characterized all the time as having a multi-instrument context.

  Parallel access to all the mounted instruments will be provided, though only one instrument per main telescope has access to the telescope beam (active instrument).

- Multi-user context

  Independently of the location of users (in the control room or at a remote location), they will be able to access any part of the whole set-up with a simple logon and configuration operation. In other words, the whole VLT system will be accessible and controllable from any single user station.

  A monitoring mode might also be important when a problem occurs, for which expert advice is needed and this can only be obtained from colleagues situated remotely (either in Santiago or in Garching). Monitoring will allow them to follow the results of tests performed and investigate how the system is working.

D *Operational requirements*

The VLT shall be operated at different levels: observing, maintenance and test level. This basically corresponds to the needs of the different users, namely observing astronomers, operations staff and software development staff.

Observing astronomers differ very much in experience and many of them are very occasional users of the system.

A system of privileges and protections shall allow operation of the different parts of the VLT at the required level, without interfering with other users.

A capacity of twenty active users at the same time is foreseen, with six additional users who can monitor the work of others.

- Service observing

  Observing, as such, is the purpose of the VLT system, but classical (interactive) observing is the least efficient way to achieve this.

  Service observing, on the other hand, means that the observing programme can be performed by someone other than the proposing astronomer.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S05SRN03

- Flexible scheduling

  Flexible scheduling means the possibility of reacting quickly to changes in weather and other conditions by allocating the optimal observing program for those conditions.

  It requires the use of service observing, as users may not be able to wait for special conditions.

- Automatic observing

  To achieve efficient service observing and flexible scheduling, it will be possible to carry out observations automatically, in accordance with pre-defined sequences of exposures, as is commonly done in space observatories.

  At the same time, one does not want to lose the advantages and the extra flexibility of ground-based astronomy. So whichever scheme is adopted to perform automatic sequences, interaction will be allowed at the desired level (for example, only on error conditions) and at any time the user shall be allowed to break a sequence.

II. VLT CONTROL SOFTWARE

The VLT control software includes the control of the main and auxiliary telescopes, the interferometer and the instruments of the VLT (18 are foreseen at the moment).

All this will be based on a distributed environment of workstations and microprocessors, as described below. The total number of microprocessors for which specific VLT software will have to be written is estimated to be in the order of 150, of about 40 different types.

The total amount of software to be produced, excluding the instruments developed by Astronomical Institutes linked to ESO is estimated to be in the order of 90-100 man-years.

A Data specifications

The final purpose of the VLT software is the acquisition of astronomical data in digital form in the most efficient way.

To achieve this, many other data concerning the telescopes and instruments and control commands will have to be exchanged between different processing units in order to set-up and control telescopes and instruments. Additionally, video and voice data are also necessary (for example, field monitors).

Control information must be transferred, typically in the form of commands and replies from users, to telescopes and instruments. Replies might contain status information and, in general, data concerning instruments and telescopes, to be stored together with the astronomical data.

B Control system distribution

The VLT system is intrinsically distributed and needs user access at any telescope and instrument. This leads naturally to a distributed structure. The implementation plan, with different telescopes going into operation at different times, demands a stepwise implementation of the control system.

When one takes into account the control needs of the various elements, every telescope is itself a distributed system where control functionality and computing power has to be provided.

All the above considerations lead to a control system design, distributed both at the telescope level and again at a higher level, to cope with the distributed architecture of telescopes and instruments.

A distributed system improves the hardware reliability of the whole set-up, as the number of cables is minimized. This has been tested on the NTT.

C Control system layout

Figure 2 gives a schematic view of the VLT control system, which ESO has in mind.

The four main telescopes, subsystems and instruments are shown in the top part at the left. The central part shows the computer and communications equipment located in the VLT control room. The bottom part shows in a schematic form the remote access facility in Garching.

The part on the right at the top shows the Coudé combined focus facility, while the Astronomical site monitor is shown on the right hand side at the bottom.

Additionally there will be a separate control room for interferometry.

D Control system architecture

Referring again to Figure 2, one finds at the bottom (near the control electronics interfacing with telescopes and instruments) the Local Control Units (LCUs) and then a network of coordinating processors (telescope and instrument processors and workstations) and central and remote workstations.

In this respect the hardware layout has a two-layer structure, but the various telescope segments are separated via bridges and gateways from the network backbone.

Functionally, a hierarchical structure with three layers can be implemented in the software, separating the local branches of the local area network from the central backbone, so that the central computers can be seen as the top layer. This offers advantages in modularity and security in the case of operations at the observing level (e.g. a command from a central workstation might go to the coordinating processor of telescope 1 and from there, be rerouted to the appropriate LCU).

At the same time at the test level the normal operational hierarchy can be by-passed and any local controller can be addressed and tested directly from a central workstation (two-layer hierarchy).

The system described:

- supports access from any workstation to all or parts of the VLT,

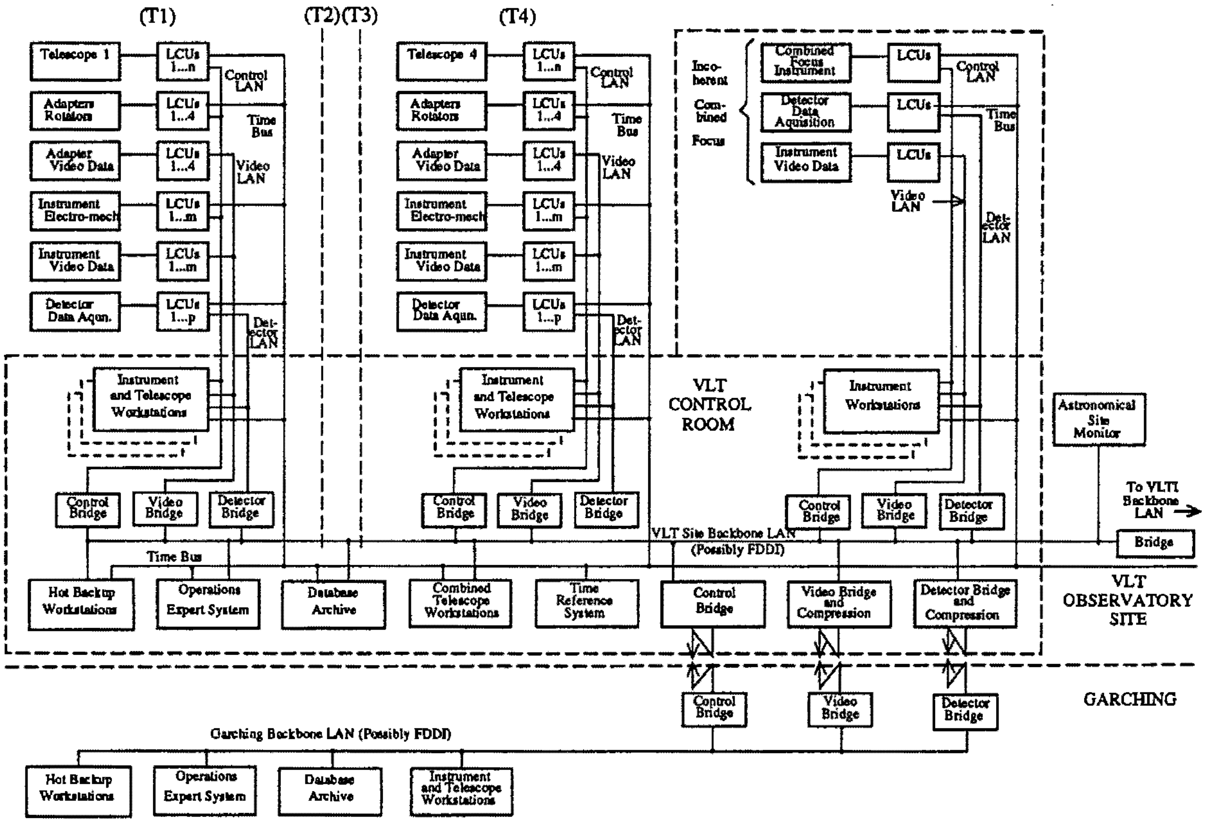


Figure 2: VLT control system

- can be implemented in steps, allowing independent development of the various instruments, and
- implements distributed real-time control at the telescope level, via the LCUs.

Different LANs have been introduced with the purpose of coping with the different requirements represented by synchronisation needs, control data, astronomical data and remote access.

E *Functional Software Structure*

The VLT software can be divided into five major functional blocks:

F *Development model*

The VLT software will be developed according to a precisely prescribed development plan and to methodologies supported by the use of CASE tools.

The purpose of this is to:

- make the development process visible
- provide maintainable software
- cope with the need for expansion of the VLT software.

G *Standards*

Standards have been selected for the computers and the system software to cope with the development phase and to allow a smooth transition to the commissioning phase when the target computers will have been chosen.

The following criteria have been applied:

- Emphasis on development and productivity requirements, including cross-support tools.
- Portability of software (target hardware independence)
- Hardware and vendor independence
- Use of industrial and de-facto standards

For the operating systems Unix System V (with Posix interfaces) has been chosen for the workstations. This is complemented by the X11 window system and OSF/Motif in the area of user interfacing. Communications portability shall be achieved by using TCP/IP and ARPA services. Command handling will be based on RPC. Commercial products are used in the area of the off-line database (Sybase) and as a CASE tool (RTEE). VxWorks is the operating system which has been chosen for the VME-based microprocessors (local control units). This product is particularly suitable for a distributed environment and is well integrated with Unix, having the necessary real-time characteristics. The software development language has been defined as being ANSI-C, with a number of restrictions applied to it. This has resulted from a comparison with ADA and C++. There are thoughts that this last could be used in some selected areas.

Communications hardware is IEEE 802.3 for the control LAN and might become FDDI for the data LAN. FDDI will also be used for the workstations backbone.

H Functional Software Structure

The VLT software can be subdivided into two categories : system software (or data acquisition and control software) and specific applications software (in this case Telescope and Instrumentation control).

Figure 3 shows the structure of the VLT software.

The system software will be described in more detail in the next sections.

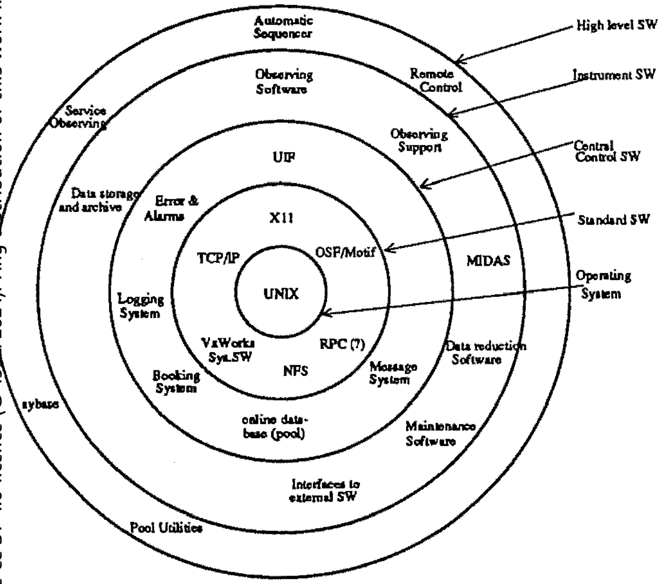


Figure 3: VLT Software Structure

I System Software Structure

This contains all the common services, which allow an open and expandable system such as the VLT to look like one homogeneous entity to the users.

In particular common (non-application-specific) software, including protocols, libraries, utilities, tools, on-line database and interfaces to the external software, user interface tools are part of this.

The system software will contain the following modules:

1. User Interface common tools

   The VLT will accomodate many user stations, locally and remotely (both in Chile and in several sites in Europe). All of these stations, no matter for which instrument or telescope they are used, will have a common "look and feel". This will be achieved by implementing a portable user interface toolkit, based on OSF/Motif, which will be used for the implementation of any particular user interface.

   The user interface will be completely decoupled from control programs, in that the values being displayed or cyclically monitored will come from an on-line database rather than being requested directly from the control programs.

   This will detach user interface issues from program development, and will allow rearrangements in this area easily.

   In this way the variable number of users and monitoring stations will not affect system performances.

2. On-line database

   The control parameters and any other information relating to the telescope, instruments and detectors must be accessible at any operation level and shall be contained in a control on-line database.

   This has to be a real-time database, with the access-critical data based in memory. Access to information in the database must be logical (for example, by name).

   The database must include mechanisms to allow the collection of data from a number of LAN nodes and their use by others. So it must have mechanisms to allow either remote access or it must support data distribution (or both).

   The database must have an interface to general database management systems (DBMS) and conversion tools from/to it via standard interfaces must exist.

   Table-driven applications can thus be easily implemented in a clear and maintainable way all over the system.

   A real-time distributed database with a hierarchical structure has been implemented by ESO for the NTT project [4] and has been ported now to Unix for the remote control of the NTT from Munich (see later remote observing).

3. Communications software

   A generalized protocol will be used across all software packages for command passing at the program-to-program communication level. Commands should have a logical nature and shall not refer to any specific physical address.

Cascade routing of messages will be supported to allow remote access.

4. Common LCU software

This includes all the common microprocessor software, contained in the LCUs, but not yet application specific.

Command passing, downloading of initialisation files, common rules to access and exchange information, synchronization methods, lists of common generic commands and boot procedures have to be incorporated in this software.

III. VLT REMOTE OBSERVING

Remote observing means that users shall be able to observe from a remote site, Santiago, Garching or even home Institutes in Europe. This is explained in detail in [5], as ESO has already been running a rather unique regular remote observation service from Munich with the Observatory in Chile, for about 4 years.

The extent to which realistic observing conditions can be reproduced depends, of course, largely on the link bandwidth available. Experience with previous telescopes shows, however, that, even with very limited bandwidths, remote observing can be implemented, provided the software is suitable for this.

Remote observing, using the VLT will be indirect. Users will have an interaction possibility, but normally via operators at the VLT or in Garching, or via the scheduler program and shall not control any part of the VLT directly.

Remote monitoring is the simplest level of remote observing. It is sometimes called 'eavesdropping'. It is a requirement for the VLT operation and will complement service observing, making it friendlier for users.

The remote observing facility in Munich will be linked, according to the current plans, with the site at Paranal with a dedicated leased satellite link at 1-2 Mbit/sec using ESO-owned antennas.

IV. CONCLUSIONS

The VLT system software is now in the analysis phase. Reuse of some components is planned, while for new software ESO is planning to take advantage of collaborations with Astronomical Institutes and industry.

There are a lot of areas in the VLT control software, which are common to other control applications in physics. The common denominator for high energy physics, fusion research and astronomy being the production of fairly high amounts of scientific data, for later analysis and also with on-line processing requirements.

Some relevant areas for possible collaboration include:

- UIF portable toolkits (although specific applications will be different)
- Command handling in distributed environments (application level protocols and internal syntaxes).
- Use of on-line distributed databases in control systems, in combination with commercial database management systems. Computer and software vendors are also starting to enter this area.

These are some of the areas in which collaboration and exchange of information can play an important role (see also [6]).

V. REFERENCES

[1] D.Enard, "The European Southern Observatory Very Large Telescope", J.Optics,vol.22,n.2,pp.33-50, 1991.

[2] L.Noethe et al., "Latest developments of active optics of the ESO NTT and the implications for the ESO VLT", in Proceedings of SPIE, vol.1542, 1991.

[3] F.Merkle et al., "Adaptive optics for the European Very Large Telescope", in Proceedings of SPIE, vol.1542,1991.

[4] G.Raffi et al., "NTT control/acquisition system as a prototype for the VLT", in Proceedings of SPIE, vol.1235, Tucson, USA, February 1990, pp.428-435.

[5] G.Raffi, "Remote Observing", in Proceedings of ESO Conference on Very Large Telescopes, Munich, Germany, March 1988, pp.1061-1072.

[6] A.Daneels, "Controls in Experimental Physics", Europhysics News, vol.22, pp.23-25, February 1991.

Distributed Control And Data Acquisition For The EUROGAM Gamma Ray Spectrometer

E C G Owen
Daresbury Laboratory
WARRINGTON WA4 4AD
UK

Abstract

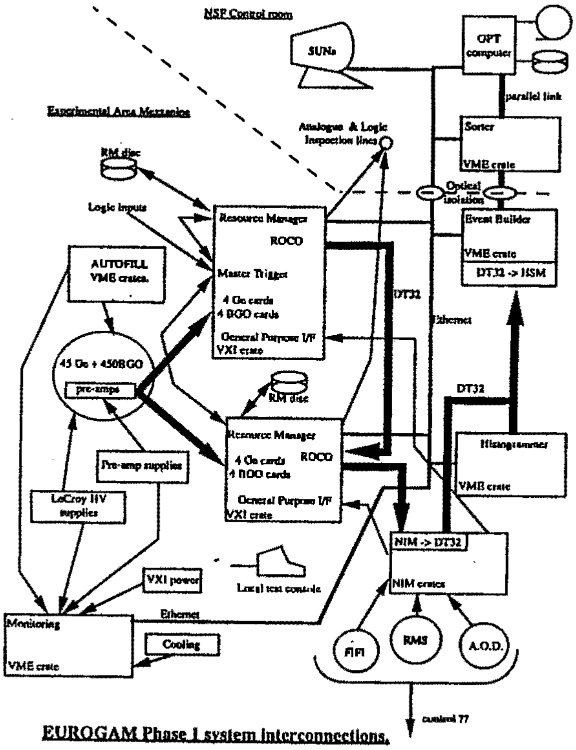
EUROGAM is an Anglo/French Gamma Ray Detector which will alternate between the Tandem Van der Graaf at Daresbury and the Vivitron at Strasbourg. Because of the need to conform to the standards of Laboratories in two different countries, and the very sensitive nature of electronics for Germanium Gamma Ray telescopes, the newly emerging VXIbus (VMEbus EXtensions for Instrumentation) was chosen as the basis for control and data acquisition. This entailed a major programme of development for both the signal processing front end modules for Germanium and Bismuth Germanate detectors, and also for the hardware and software management of resources from within the VXI environment. The paper will concentrate mainly on the latter areas.

I. INTRODUCTION

EUROGAM is a high resolution gamma-ray detector which is being constructed to answer the many exciting questions raised through recent discoveries in Nuclear Physics. Phase I of the EUROGAM array provides 45 Germanium (Ge) detectors with surrounding Bismuth Germanate (BGO) detectors being used to provide the suppression shields. A second phase is planned which will provide a 70 detector system. EUROGAM is a joint development project between Institut National de Physique Nucléaire et de Physique les Particules (IN2P3), France, and the Science and Engineering Research Council (SERC), UK. The array is expected to come into operation at the Nuclear Structure Facility at Daresbury Laboratory, UK, early in 1992 and will run for one year following which it will be transferred to France to operate on the new Vivitron accelerator which is being commissioned at Centre de Recherches Nucléaire, Strasbourg (CRNS).

II. TECHNOLOGY

To provide data acquisition, control and monitoring functions for EUROGAM, a mixed solution based on VXI, VME and UNIX workstations has been chosen with the overall system structure being arranged in a distributed computing architecture. Figure 1 shows the layout of the system and its component parts.



EUROGAM Phase 1 system interconnections.

FIGURE 1.

VXI is used to front-end the system and has enough electric and magnetic shielding to allow the treatment of analogue signals to very low noise levels This permits a highly integrated approach to be taken in the electronics units designed for the processing of pulses from detector channels. VXI also minimises the inter-unit cable connections required and hence increases reliability. VXI also provides a full implementation of a 32 bit VME bus for general control of cards in the crate, read-out capability and multi-master processor access. The enhanced features contained in the VXI standard allow user definition of specific lines to suit individual system requirements, trigger implementation, clock facilities, and if higher speed data read-out is needed in the future, a 32 bit local bus is available.

Components of the system not requiring a very low noise environment are implemented in a number of standard VME crates for economy, commercial availability of many modules, and efficient use of space. Functions implemented in this form are Histogramming, Event Building, On-Line Sorting of Data, High Voltage Control and Autofill Systems for detector cooling. The overall system is set-up, controlled and monitored from UNIX workstations via an Ethernet LAN. A separate high speed data path is provided for data acquisition. Users of the system can also use the same workstation for visualisation of on-line and off-line data. VXworks has been used for the real time software components of the system and runs in all the VME/VXI based processors.

III. VXI SYSTEMS

The VXI front-end crates are dedicated to processing of analogue pulses from the detectors. Special components have been designed and manufactured for the Ge and BGO interface boards which accept the analogue output pulses from detectors and provide all the signal processing required to produce a digital output data stream which can be accessed over the backplane. A 6 channel Ge board is available and this is matched by a BGO card which provides signal conditioning, event vetoing and read-out from the suppression shields surrounding the Ge detectors. All adjustable parameters on the Ge and BGO cards are under control of the software via read and write access to registers over VME bus. Local trigger facilities are also provided on each board. Each VXI crate is controlled by a Resource Manager (RM) unit which also contains the Slot 0 module functions. The RM is modular in its construction and allows the integration of commercial VME processor boards offering an on-board Ethernet network interface with the 3 other RM boards which provide VXI/VME interfacing. Thus the architecture of the RM can be updated or tailored to meet other experimental requirements without complete re-design. The current RM has been developed by electronics engineers working on the EUROGAM project and engineering staff from Struck in Germany. The company are responsible for the manufacturing of this unit. Read-out for each crate is provided by a Read-out Controller (ROCO) which shares mastership of the VME bus with the Resource Manager. The ROCO collects data from each instrumentation card participating in a multi-parameter event and forms a data block for transmission via its output port which is a 32 bit ECL bus (DT32) capable of rates up to 4 Mbytes/sec. The DT32 bus is chained to all ROCOs in the system and read-out is performed sequentially from each VXI crate. A Master Trigger unit is required for the data acquisition system and this is also designed in VXI format and housed in one of the front-end crates. A small amount of trigger logic is associated with each detector channel and this is provided by the local trigger facilities on-board each instrumentation card. Master Triggers are produced when one of several pre-programmed multiplicity levels is reached (ie, a number of detectors involved in multiparameter event). This is done by summing current outputs provided by the local trigger of each active channel and combining this with logic inputs provided directly to the Master Trigger. The trigger unit is programmable and software access is again via registers on the VME bus.

IV. HISTOGRAMMER

The ability to histogram parameters from individual detector channels is a requirement of the system and this is achieved by inserting a Histogrammer into the DT32 chain. This is a single crate VME system which consists of a special interface together with commercially available memory and processor modules. The special interface unit spies on the data passing over the DT32 but does not participate in controlling the flow of data from the ROCO. Data to be histogrammed is loaded into a FIFO which acts as a de-randomising buffer between the 10 MHz burst rate on the DT32 and the average rate at which histogram updates are required. A software loadable look-up table is used to compare the address fields of each parameter passing over the DT32 with internal values to see if this parameter is to be histogrammed. If a match is found the look-up table also provides the base address of the area of memory assigned to this parameter. By summing the base address and current data value of the parameter the correct memory location is then incremented over VSBbus. Histograms are read out over the VMEbus and shipped by the crate control processor over the set-up and control Ethernet for visualisation at a user workstation. Such histograms are generally used to check the quality of data from an experiment and to monitor detector performance. If data rates exceed the capability of one histogrammer unit additional systems can be inserted into the DT32 bus and assigned different groups of parameters to monitor.

V. EVENT BUILDER

The Event Builder (EB) is a multiprocessor VME system which is built entirely of commercially available modules, except for a small interface board (ROHSI) which terminates the DT32 bus and passes data to one or more CES HSM8170 units which are controlled by FIC8232CPU. The main task of the EB is to form self-contained events from the raw data stream for onward shipment, further processing and/or storage. The EB can also carry out simple functions such as individual/shared suppression filters, event filtering and simple arithmetic operations. The system must be capable of processing 100% of the data in real-time and it is therefore equipped with a number of MVME165 processors (4 initially) which are assigned to event processing and operate in parallel on the data stream. The EB is initially targeted to process about 1 Mbyte/sec of raw data but is expandable by the addition of further hardware modules. The output stream of formatted events is via a dedicated fibre optic link driven by a CES FIC8232 and CES ODL8142. The fibre optic link and its drive units are capable of operation at rates up to 6 Mbytes/sec. Control of the EB is via the Ethernet LAN and as the unit is fully programmable other software functions can be added as experience of user requirements is gained during operation of the array.

VI. DATA ROUTER AND SORTER

The Data Router and Sorter (DRS) is a VME system which sits at the remote end (ie outside the experimental area) of the fibre optic link driven by the EB. In addition to providing a high speed data path the optical link also provides an excellent means of isolating electrical noise and potential earth loop problems in the computing zone from the more sensitive experimental area.

Although contained in the same crate the functions of routing and sorting are to be seen as completely separate facilities. The Router forms part of the data acquisition path whereas passing data to the Sorter is optional and to be selected by the user.

The optical link is received by an identical pair of units to those in the EB (FIC8232/ODL8142). The Router software then allows all or some percentage of data to be sent to the Sorter or alternatively directly written to tape; if required, both paths can be accommodated in parallel. The Data Sorter consists of a number of MVME165 processors which are controlled and supervised by an MVME147 unit which also provides general network access to the DRS via its Ethernet port. The task of the Data Sorter is to provide on-line analysis of the data. Multi-dimensional histograms are built in global memory and can then be made available to the user for viewing at a workstation. The main data stream from the DRS is output via a parallel interface to the resource computer.

VII. RESOURCE COMPUTER

The Resource Computer which provides tape and disc drive facilities to the data acquisition system is a GEC4190 machine, one of a range of computers which have been used at Daresbury for several years with great success. Tape server software has been written for the GEC4190 which allows the event by event data received from the DRS to be written to one or more Exabyte 8500 drives. The server software supports different data streams and it is the responsibility of the user to select the event type to be assigned to a particular stream. At a later stage of the project a UNIX based system will be developed to provide tape and disc resources.

VIII. HIGH VOLTAGE CONTROL

The High Voltage (HV) supplies required for the detectors are provided by LeCroy 1440 high voltage units which are controlled and monitored via a LeCroy 1131 interface board and controlled by a VME processor unit connected to the Ethernet. The software allows interrogation of each HV mainframe to find unit type, the voltage setting for each channel, ramp rate, and current limits; it will also continuously monitor all channels to check for any irregularities.

A hardware interlock to the Autofill System has been provided so that any failure of the cooling system to an individual detector will cause shut-down of the appropriate HV channel. This condition is detected by the control task and reported back to the user. A local database of the current set-up state of all HV channels is maintained and is available to the user over the network for viewing and/or storage for use on a future experiment. The user interface to the HV system will be from the UNIX workstation.

IX. AUTOFILL

It is necessary for the Ge detectors in the EUROGAM array to be kept at liquid nitrogen temperature during operation and data taking. This is accomplished by means of an Autofill System which regulates a series of solenoid valves according to the cooling requirements of each detector as indicated by temperature monitoring. A VME system with local processing capability has been provided to control the operation and much of the controlling software runs in stand-alone mode. The VME system has a network interface onto the control Ethernet thus allowing monitoring information and error messages to be passed to remote terminals for viewing. The Autofill equipment has been designed with an optional manual mode of operation and as indicated in the section on HV control there is a hardware interlock to ensure the voltage supply is removed if the temperature of the detector exceeds its limits.

X. ENVIRONMENTAL MONITORING

To ensure the quality of the environment in which EUROGAM operates, the data acquisition system is built into water cooled racking systems. To prevent damage under fault conditions and to provide early detection of potential problems, the systems are monitored for temperature, fan failure, power supplies, etc. This information is fed back to the control system via VME interfaces which are accessed over the network.

XI. UNIX WORKSTATIONS

Networked workstations provide the user, and where appropriate, engineering support staff with an interface to the whole of the EUROGAM control, data acquisition and data analysis system. This is achieved by using a window system based on X11 which gives the user access to a series of purpose designed windows which provide visualisation and control to various areas of the system. The workstations also provide a programming environment for users in which they can develop their own software tools for analysis. Menus are provided to set up EUROGAM to meet individual experimental requirements and the current configuration status can be sorted for future use. Access to the system is not restricted to any single workstation and provides facilities for groups of users to work in parallel. Security features are built into the network system to prevent access to areas of the array software and hardware facilities which could be damaged in error.

XII. SOFTWARE ENVIRONMENT

The individual hardware components of the system have been described and form a networked set of distributed computing elements. It is the software environment which provides the integrated approach to EUROGAM control and data acquisition and offers the user complete control of his or her experiment, from set up and monitoring, through acquisition to analysis functions, all controlled from a UNIX workstation offering graphical menus in multiple windows to meet all general requirements. The software has been designed so that it is modular in its architecture thus providing for easier maintenance and the development of additional facilities in the future.

Whilst UNIX workstations were selected to provide the user interface, UNIX is not considered an appropriate operating system for the real time elements of the software which run in the VXI and VME processors and which provide various functions in the EUROGAM architecture. VxWorks was chosen as the real-time kernel and has proved to be very efficient in both its ease of development and run-time performance.

The software is designed around the client/server mode of working and Ethernet provides communications to all areas of the distributed system. Four main server programs have been developed to provide access to all data acquisition resources, these are a register server, spectrum server, tape server, and message/error logger. The main path for data from the experiment is kept separate and is as described in the hardware section. This path is not used for software commands and is uni-directional. Figure 2 (overleaf) shows the overall software structure and how it relates to the hardware components.

XIII. SOFTWARE COMPONENTS

Communications

Common network software is used to access all components of the system and Ethernet is used as the carrier LAN. CPUs which are attached to the Ethernet can also act as gateways for communication with other processors in the same crate using the backplane as a network.

The protocols used for communication are:

1. NSF (Network File System).
2. RPC (Remote Procedure Calls).
3. XDR (eXternal Data Representation).
4. UDP/IP (User Datagram Protocol/Internet Protocol.)
5. IP (Internet Router).

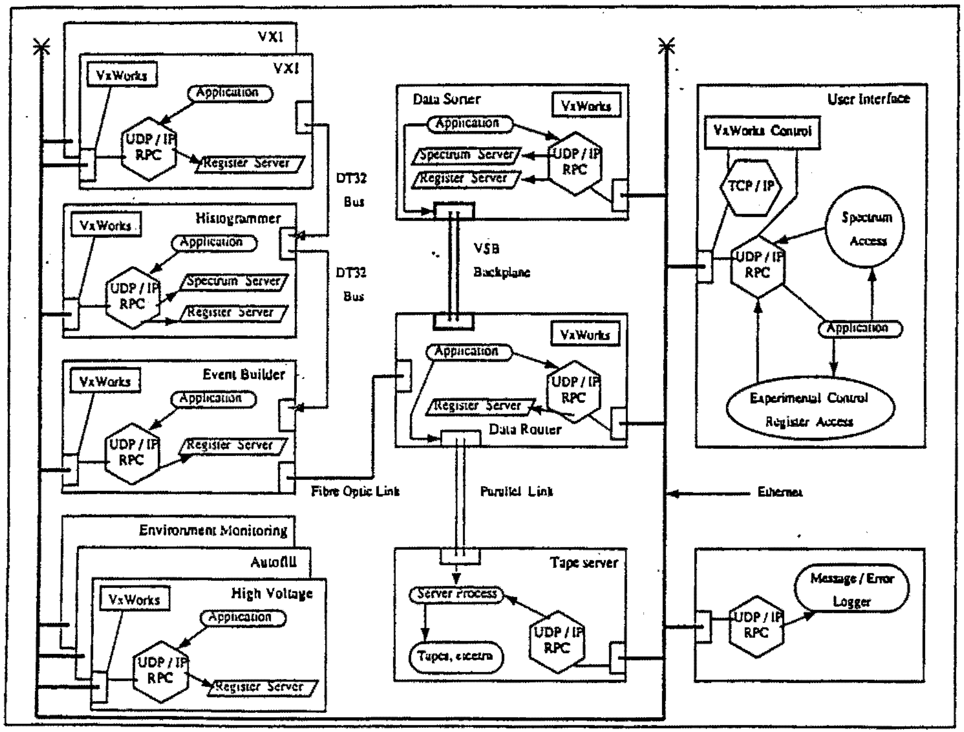


FIGURE 2.

Communications between clients and servers use RPC to pass requests for control, monitoring and read-out functions. Fixed UDP/IP ports are used by servers to avoid delays which the use of dynamic ports would involve. To offer facilities which are more suited to the data acquisition and control environment the standard Sun Microsystems RPC library has been extended to provide:

1. Asynchronous RPC to improve response to user interface requests.
2. Multiple RPC requests to different servers.
3. Variable RPC timeouts for specific applications.

XIV. VXWORKS

The decision to run VxWorks as the real-time kernel was based on the following criteria:

1. Offers a multi-tasking environment designed to meet the need of real-time control and data acquisition.
2. Software is designed and developed using UNIX workstations and is UNIX source compatible.
3. BSD networking available.
4. Performs well in the areas of task scheduling and interrupt handling.
5. Remote debugging facilities are available.
6. Support is provided for a wide range of manufacturers products

XV. REGISTER SERVER

The Register Server allows client programs to communicate with server processes anywhere within the EUROGAM system. It is a general purpose communications package which can be tailored to fit a particular application environment When the system is initialised, each server process is loaded with the application specific processes it requires to carry out its designated task which can be viewed as a set, or, in some cases, sequence of writable and readable registers within the VXI and VME systems for which it is responsible. Registers are defined dynamically by clients using the Register Server protocol and specify the local procedure to be invoked when the register is subsequently accessed. The Register Server also provides for data to be passed between client processes and server applications.

The Register Server Primitives are: a) Read, b) Write, c) Initialise, d) Read Attribute, e) Write Attribute. Attributes are additional pieces of information required by application procedures, for example the offset address of a register in an individual VXI or VME card. For reading and/or writing sequences of registers wild-card techniques have been implemented based on the register name. This allows many registers to be accessed with a single RPC request.

XVI. SPECTRUM SERVER

The Spectrum Server is responsible for the handling of all spectra and provides access to data in the Histogrammer and Sorter units and is also able to access off-line spectra held on disc. The server controls and allocates memory in the Histogrammer as required to meet user demand and will pass the address of an individual memory area assigned to a parameter to be histogrammed to the Histogrammer look-up table via the client workstation using the Register Server. In response to user requests spectra can be named and created and then read back for on-line display at the workstation or for off-line storage.

XVII. TAPE SERVER

As previously mentioned in the section on the Resource Computer the server software provides access to tape storage devices. Data striping has been implemented to permit data to be written to several units in parallel when higher data rates are required than can be accommodated by one drive. Alternatively the same data stream can be written to multiple tapes simultaneously which is a useful feature when several users in an experimental collaboration wish for their own copy to take to their home institute.

XVIII. MESSAGE & ERROR LOGGER

When problems or errors occur in the system a report is passed to this process. Such items would be HV malfunction, Autofill problems or an environmental failure. Clients programmes can declare that they are interested in receiving messages from a particular area of the system and these message types are then forwarded to the client in addition to being logged by the server.

XIX. OTHER SOFTWARE

EUROGAM provides facilities such as Event Building and Data Sorting described in the hardware section. These functions are controlled and accessed in the same way as other VXI and VME devices and use the VxWorks kernel. In order to perform their designated scientific function each system has specific application code which is tailored to meet experimental needs. This software is not described in the paper. To provide the graphical scenes required for the user interface the Open Look Window Manager is used together with DEV Guide to provide the development environment. To complete the software system several small application routines are provided to meet specific requirements, these routines use the software infrastructure described.

XX. ACKNOWLEDGEMENTS

The EUROGAM project is a funded by IN2P3 in France and by SERC in the UK. The author wishes to acknowledge and thank the members of hardware and software teams in France and the UK who have been responsible for the development and production of individual items referred to in this paper, also the physicists who have provided discussion and support for the control and data acquisition project. Special thanks to V F E Pucknell for assistance with the software section.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S05SRN05

A Control System of the Nobeyama Millimeter Array

Koh-Ichiro Morita, Toshikazu Takahashi, Hiroyuki Iwashita,
and Tomio Kanzawa
Nobeyama Radio Observatory, National Astronomical Observatory
Minamimaki, Minamisaku, Nagano 384-13, JAPAN

Abstract

We have developed a control system of the Nobeyama Millimeter Array which is a radio interferometer for astronomical observations at millimeter wavelengths. The system consists of three sub-systems (MANAGER, ENGINE, and STATUS CONTROLLER). Observers conduct their observations with MANAGER sub-system, which run on a UNIX workstation. ENGINE is a rigid system on an IBM compatible mainframe. It controls the accurate tracking of astronomical radio objects, and acquires a large amount of observed data from a receiver backend. STATUS CONTROLLER consists of several personal computers which control and monitor the receiver system. These sub-systems are connected with an ethernet.

1 INTRODUCTION

The Nobeyama Millimeter Array (NMA) [1] is a radio interferometer for astronomical observations (Figure 1). The main purpose of the NMA is the high spatial resolution imaging of celestial objects at millimeter wavelengths.

The array has five 10-m diameter antennas which can be moved to various stations along two rail tracks of about 600 m. Averaged surface accuracy for five antennas is 71 μm rms. Each antenna has an Alt-Azimutial mount and is equipped with SIS receivers [2] at 2.6 mm and 2.0 mm wavelengths which contain important molecular spectral lines. The maximum spatial resolution at 2.6 mm wavelength is about 1″. The receiver backend is a 320 MHz FFT spectro-correlator with 1024 frequency channels per correlation, called as the Nobeyama FX [3].

The most important requirement for a control system of such an interferometer for astronomical observations is to track celestial objects accurately [4]. For an interferometer, the tracking control is much more complicated than that for a single dish telescope. To satisfy this requirement, we constructed a centralized control system for the NMA on an IBM compatible mainframe [5]. However, since the softwares of the system were very rigid, there were some drawbacks in it.

To overcome these drawbacks, we have developed a new control system for the NMA. It is a distributed system, based on a UNIX workstation, an IBM compatible mainframe, and personal computers connected with an ethernet. In this report, we will describe the concept and the structure of the new system.

Figure 1: The Nobeyama Millimeter Array.

2 CONTROL FOR OBSERVATIONS WITH THE NMA

2.1 Tracking

Normal observing mode of this array is aperture synthesis[6]. Figure 2 shows a schematic diagram of aperture synthesis observation with two element radio interferometer. A correlator multiplies two radio signals from each antenna and averages the product. When antenna tracking to a celestial object and compensation of optical path difference between two antennas (delay tracking) are done accurately, the correlator output is equal to a spatial Fourier component of the brightness distribution at a projected baseline (u, v). After the synthesis observations at many different baselines, the brightness distribution can be estimated with inverse Fourier transform.

As shown in Figure 3, we use a heterodyne system for a frontend receiver and frequency of the input signal to the delay tracking system is lower than receiving frequency.

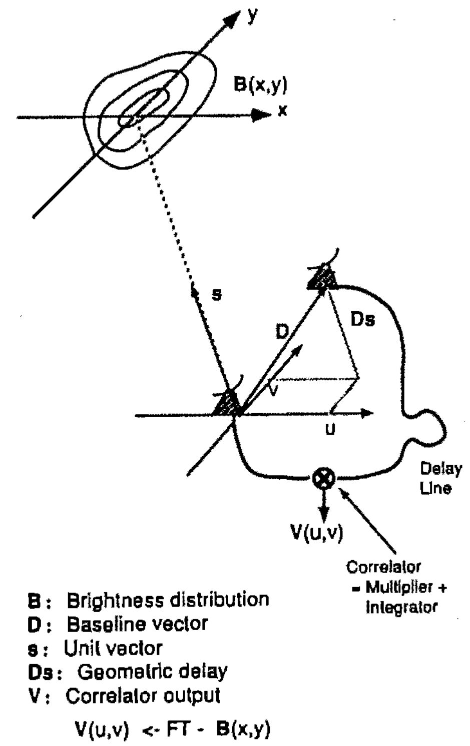


Figure 2: Aperture synthesis observation with two element interferometer.

Since, in such a case, the delay tracking cannot compensate the phase difference due to the path difference, exactly, it is necessary to compensate residual phase by controling the phase of LO with a fringe rotator (phase tracking).

Therefore, for aperture synthesis observations with the NMA, we must control the antenna tracking, the delay tracking, and the phase tracking. Since the resolution of the delay tracking system is finite, control sequences of the delay tracking and the phase tracking are very complicated. These control accuracies affect the qualities of radio images obtained with the aperture synthesis observations[4].

2.2 Tuning of the Receiver System

The receiver system of the NMA consists of SIS mixers, LO oscillators, IF amplifiers, and a reference signal synthesizer. Before each observation, observers must optimize many parameters of each component of the receiver system. This task which is called "tuning" is very important for sensitivities. At higher frequencies, the tuning becomes much more important and its sequence is very complicated. These know-hows of manual tuning must be programmed for each device in the control software.

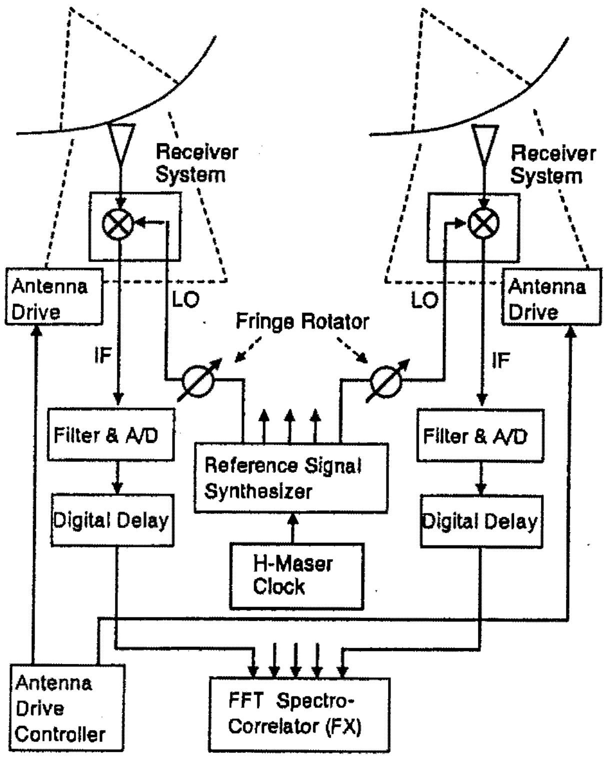


Figure 3: Hardware system of the NMA.

2.3 Operational requirements

Generally, astronomical observations with radio telescopes are serieses of many short observations of different celetial objects. In typical aperture synthesis observations with the NMA, of which duration is 6 ~ 10 hours, a 10 ~ 15 minutes observation of standard objects is taken every 30 ~ 60 minutes in order to monitor complex gain variation of the array. Such observing sequences depend on observational requirements of observers. Thus, it is desireble that the control system provides an environment where observers can easily realize their observing sequences.

Since we have many observers who are inexperienced in the NMA, a friendly human interface is very important for the control system. Quick display of backend data, observing parameter menus, and multi windows are very effective for smooth operations.

3 OLD CONTROL SYSTEM

The old control system of the NMA was a centralized system, which ran on an IBM compatible mainframe and two mini-computers. Main control tasks (sequence control of observing modes, tracking control, data acquisition, human interfaces, etc.) ran on the mainframe. A terminal of the mainframe was used as a telescope operator console. The mini-computers managed communication between the mainframe and each device of the array which was connected with the mini-computers via GPIB. Hydrogen maser clock was supplied to the mini-computers so that tracking parameters were sent to the tracking devices at the exact time. With this system, we realized high tracking accuracy and high reliability of array control [5].

However, since softwares of the system were complicated and not flexible, there were following drawbacks in it:

1. To add functions for new devices takes very long time.
2. It is difficult for observers to develop control programs for their own observing sequences.

In particular, since the progress of the receiver system is very rapid, it was difficult to develop control programs for various receiver tuning methods. Besides these drawbacks, it is hard to find out good graphical user interfaces (GUI's) for IBM compatible mainframe. To overcome these problems, we have developed a new control system of the NMA.

4 NEW CONTROL SYSTEM

The new control system of the NMA, which is shown in Figure 4, is divided into three sub-systems (MANAGER, ENGINE, and STATUS CONTROLLER). MANAGER subsystem as a telescope operator console provides the human interface. A UNIX workstation has been introduced in this sub-system, because of its high graphic capabilities and multi-window environment. ENGINE sub-system tracks astronomical objects and acquire the data from the backend. Tuning of the receiver system is realized with STATUS CONTROLLER sub-system. Several personal computers are used for this sub-system, because of their simplicity of programming and reduced costs. These subsystems are connected with an ethernet.

4.1 MANAGER

MANAGER runs on a UNIX workstation, Fujitsu S-4/370 (SPARC 370 compatible) with a main memory of 32 MByte. Functions of MANAGER are as follows:

1. Making command tables (OBSERVATION BOOK) for various observing sequences and sending them to ENGINE.
2. Communicate with ENGINE by using of a SOCKET interface.
3. Providing flexible GUI based on the X-Window to observers.
4. Acquisition of the monitoring data from each device.
5. Displaying the backend data quickly.

Figure 5 shows the software structure of MANAGER. There are several standard programs (OBS.BOOK generators) which make standard OBSERVATION BOOKs in this sub-system. Since these programs are not directly connected with real-time control tasks, it is easy for observers to make new ones for their observational requirements. The SOCKET interface system between the workstation and the mainframe was developed by Fujitsu.

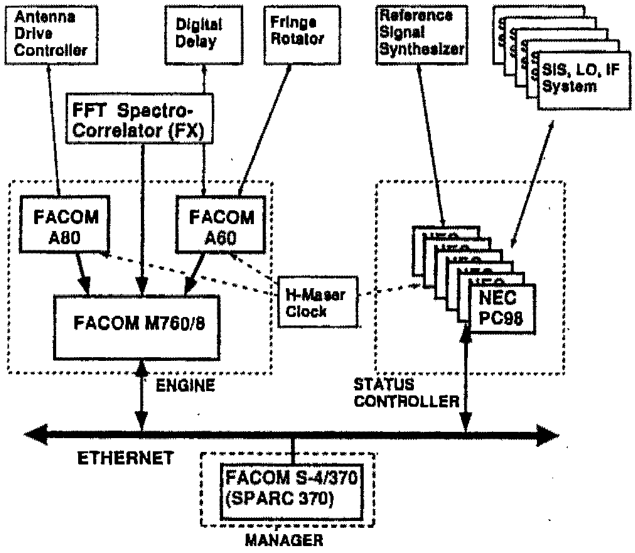


Figure 4: New control system of the NMA.

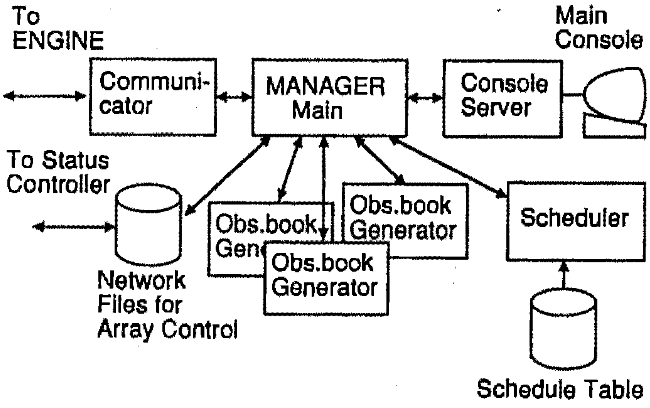


Figure 5: MANAGER.

4.2 ENGINE

ENGINE runs on an IBM compatible mainframe (Fujitsu M760) and two mini-computers (Fujitsu A80 and A60). This hardware structure is the same as that of the previous control system. The followings are functions of ENGINE:

1. Executing observing sequences in an OBSERVATION BOOK from MANAGER.

2. Calculating accurate tracking parameters for every 400 ms.
3. Sending the paramters to each device at the exact time.
4. Acquiring the data from the backend and applying the realtime correction to these data by using of control parameters.

Software structure of ENGINE is shown in Figure 6. There are three main tasks on M760, which share a common memory area to communicate each other quickly. This sub-system is very rigid so that it is possible to realize the accurate tracking control and data acquisition without data losses. Softwares of tracking control and data acquisition are simple and compact compared to those of the previous system.

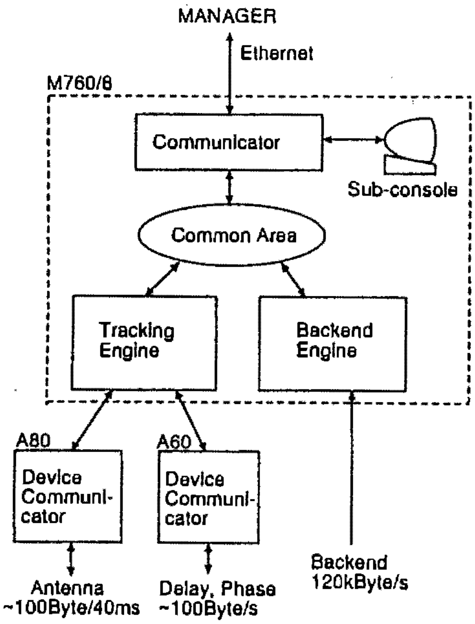


Figure 6: ENGINE.

4.3 STATUS CONTROLLER

Functions of this sub-system are as follows:

1. Controling and monitoring many components of the receiver system.
2. Communicating with MANAGER through NFS.

This sub-system consists of 6 MS-DOS 80286-based personal computers (NEC PC9801). Personal computers with MS-DOS do not have a multi-tasking environment. However, programming is so easy on each personal computer that duration for the software developement is very short. Actually, a receiver engineer wrote the software for the tuning by himself. Thus, the software can use his know-how of the receiver system.

5 SUMMARY

We have developed a new control system of the Nobeyama Millimeter Array. While the old system was a centralized one, we have constructed a distributed control system which consists of three sub-systems. Thus, the new system will provide a friendly GUI's for observers and flexible viroments for adding new functions. The developement has almost completed and has been be used for observations from the end of 1991.

References

[1] M. Ishiguro et al., "The Nobeyama Millimeter-Wave Interferometer", in "Instrumentation and Techniques for Radio Astronomy", ed. P. Goldsmith, IEEE Press, 1988, pp.151-155.

[2] R. Kawabe et al., "A Dual-Frequency 40/100 GHz SIS Receiver for the Nobeyama Millimeter-Wave Array", in *the 3rd Asia-Pacific Microwave Conference Proceedings, Tokyo*, 1990, pp.217-220.

[3] Y. Chikada et al., "A 6 × 320-MHz 1024-Channel FFT Cross-Spectrum Analyzer for Radio Astronomy", *Proc. IEEE*, vol. 75, pp.1203-1208, 1987.

[4] K.-I. Morita et al., "Tracking Accuracy of Astronomical Objects in the Nobeyama Millimeter Array", *Trans. IEICE Japan*, vol. J73-B-II, pp.41-50, 1990.

[5] K.-I. Morita et el., "Control System of the Nobeyama Millimeter-wave Array", *University of Tokyo, Tokyo Astronomical Observatory Report* vol. 21, pp.82-100, 1987.

[6] A. R. Thompson, J. M. Moran, and G. W. Swenson, Jr., "Interferometry and Synthesis in Radio Astronomy", New York: John Wiley & Sons, 1986.

Present Status of the JT-60 Control System

T. Kimura
Naka Fusion Research Establishment
Japan Atomic Energy Research Institute
801-1 Mukohyama, Naka-machi, Naka-gun, Ibaraki-ken 311-01 Japan

Abstract

The present status of the control system for a large fusion device of the JT-60 upgrade tokamak is reported including its original design concept, the progress of the system in the past five-year operation and modification for the upgrade. The control system has the features of hierarchical structure, computer control, adoption of CAMAC interfaces and protective interlock by both software and hard-wired systems. Plant monitoring and control are performed by an efficient data communication via CAMAC highways. Sequential discharge control of is executed by a combination of computers and a timing system. A plasma feedback control system with fast 32-bit microprocessors and a man/machine interface with modern workstations have been newly developed for the operation of the JT-60 upgrade.

1. INTRODUCTION

The JT-60 tokamak is a large scale fusion experimental device for the study of magnetically confined plasmas near the thermal break-even condition. Since the first plasma obtained in April 1985, studies on impurity and particle control, confinement of high-power heated plasmas and steady state operation by radio-frequency wave current drive and production of the bootstrap current discharges were performed. The JT-60 tokamak has been upgraded in order to push forward these fusion researches conducted in the past five-year operation [1]. In the JT-60 upgrade (JT-60U) we can perform deuterium discharges with plasma current up to 6 MA and additional heating power of 50 MW. Throughout these investigations we intend to obtain physical and technological databases for the next-step machines.

This paper reports the present status of the JT-60 control system including its original design concepts, progress of the system in the past five-year operation and modification for the upgrade.

2. REQUIREMENTS AND DESIGN PHILOSOPHY

Since the JT-60 tokamak is a large-scale device with respect to the number of components, their occupied space, the amount of electric power consumption, etc. Since intrinsically unstable plasmas are produced and maintained, the control system is required to have high reliability and high speed. Besides these features it has to possess the characteristics of flexibility, expansibility and safety. Hence, the JT-60 control system was designed and fabricated with the following features [2]:

(1) Hierarchical structure

JT-60U consists of more than ten subsystems such as a vacuum pumping system, magnet power supplies and plasma heating apparatus. These subsystems have to be separately operated in their preparatory stages before plasma operation. Moreover, they have to be organized into one system to perform plasma operation. Hence, the JT-60 control system has hierarchical structure of its central control system named ZENKEI and subsystem controllers.

(2) Computer control

Computer control was introduced for the operation of the JT-60 tokamak. Eight minicomputers and about a hundred microcomputers are used for a wide variety of control functions from fast feedback control of plasma discharge to handling of a large amount of monitoring and control data. In addition to 16-bit computers in the original control system, advanced 32-bit microprocessors and workstations have been newly introduced in the control system.

(3) CAMAC interfaces

Since fabrication for each subsystem was contracted to industry separately, various kinds of standards for both hardware and software were decided as JT-60 standards at Japan Atomic Energy Research Institute (JAERI). CAMAC standards were adopted for input/output signals from sensors /to actuators and for data transfer between the computers.

(4) Protective interlock by both software and hard-wired systems

The safety philosophy was established by taking into account certain key requirements in the design of the JT-60 control system. One of the most important requirements is personnel safety. This stems from use of high electric voltage, high magnetic field and possible radiation in the JT-60 tokamak. Hence, the protective interlock system with hard-wired relay logic backs up the computer system from the view point of reliability and safety. Moreover, the concept of precaution and protection was introduced in the system design.

3. SYSTEM CONFIGURATION

3. 1 Control Configuration

The control concept of the JT-60 tokamak can be classified into two categories. One is control for plant control and monitoring and the other is for plasma control via actuators used for tokamak discharge.

The concept of plant support control is similar to that in other large-scale facilities such as manufacturing and power plants. Plant support control is provided for controlling conditions of machines and monitoring their status. Included are on systems for operation-mode control, alarm monitoring, emergency control, protective interlock, etc. These control systems are supported by a complex system of computers and CAMAC systems named the "plant support and monitoring" system .

Sequence control of the actuators and fast feedback control of plasmas are included in the plasma discharge control. This control concept stems from a distinctive feature in the tokamak operation where plasma discharge with duration of a few tens of seconds is executed every few tens of minutes. The plasma discharge control is executed via three complex systems of computers and CAMAC systems named the "discharge control", "real-time control" and "feedback control" systems. Timing systems are also used for supplying trigger pulses and clock pulses in the discharge sequence control and the real-time and feedback plasma control.

The computer configuration of the JT-60 control system is shown in Fig. 1.

3.2 ZENKEI Computer System and Changes of Its Configuration

The computer system for ZENKEI was originally composed of seven 16-bit minicomputers of HIDIC-80Es (Hitachi Ltd.) having main memories of 320 to 448 KW. The cycle time of CPU is 0.48 µsec. Five of them, having a shared memory of 128 KW, were located in the JT-60 computer room. The rest, having a shared memory of 64 KW, were in a local control room of the rectifier building. The first five CPUs are used for the plant support and monitoring (Ia), the discharge control (Ib), the real-time plasma control (Ib^R) and their man/machine interface (Ia^M and Ib^M). The latter two (IIb) were used for making a feedback control loop for plasma equilibrium control in combination with direct digital controllers (DDCs) in the poloidal field coil power supply. The IIb computer was connected to the Ib computer through an optical linkage bus named "data freeway". Since the ZENKEI

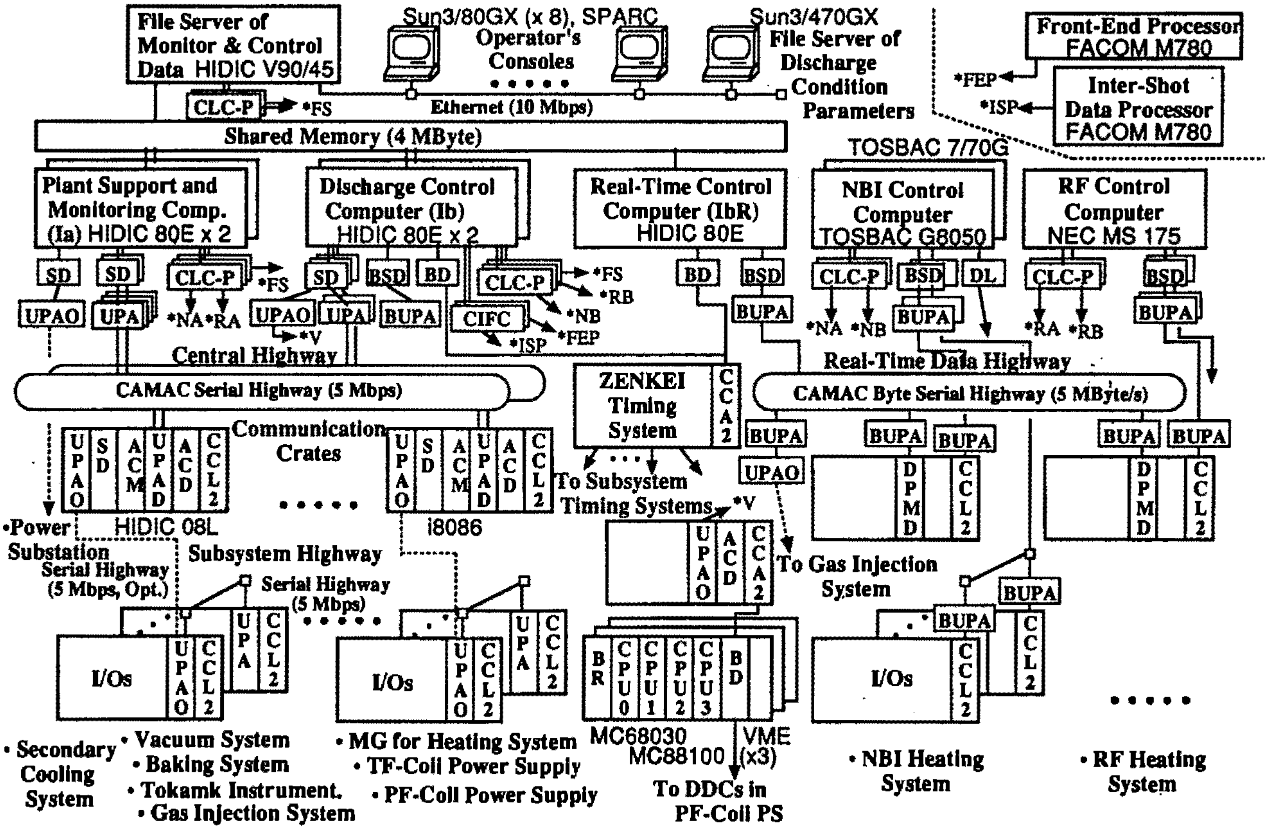


Fig. 1 Computer Configuration of the JT-60 Control System

computer system is a multi-bus system with I/O buses that are controlled through both of PCMA (Processor Controlled Memory Access) and DMA (Direct Memory Access) channels, the CPUs can share their peripherals of disks, printers, etc. A real-time operating system named "Process Monitor System-M (PMS-M)" is employed in this multi-computer system. The PMS-M has the functions of inter-CPU watching, control of tasks under other CPUs, control of I/Os shared by multi-CPUs, etc.

The configuration of the ZENKEI computer system has been changed along with the progress of the past five-year operation. The feedback control was originally performed by one of HIDIC-80Es and the other is for back-up. Since the requirements for the control system were increased, the IIb system was, then, modified to be a pipelined system by using the two HIDIC-80Es. Moreover, a preprocessor system with digital signal processors (DSPs) was added to the pipelined system. Coping with the requirements for faster and more accurate control in the JT-60 upgrade, the minicomputers in the IIb system have been superseded by a VME-based multiprocessor system equipped with 32-bit RISC processors. Modern workstations connected via an Ethernet network have been introduced for improving the ZENKEI man/machine interface, which previously was implemented on two of the minicomputers HIDIC-80Es and consoles with dedicated character and graphic terminals.

4. DATA COMMUNICATION AND CAMAC SYSTEM

4.1 JT-60 Monitoring and Control Data

The JT-60 control system handles a large amount of data for plant support and monitoring and for discharge and plasma control. The plant support and monitoring system must send device status data for each subsystem to the central control computer every 5 sec to 1 minute. The number of monitoring points amounts to about 7500 analog and 9500 digital inputs. Second, it must possess sufficient data transfer capability so that operators are able to watch the discharge results and quickly decide the conditions for the next shot. The original discharge data amounted to about 3 MB per shot (10 sec discharge duration) every 10 minute in the standard operation scheme. At present, the data amounts to about 8 MB due to increase of the discharge duration corresponding to the upgraded JT-60 tokamak. Thirdly, it must exchange several tens of control commands and status data with each subsystem every 10 msec for the real-time control and every 1 msec (250 μsec for the upgrade) for the feedback control of plasma position and shape.

4.2 Communication Systems

As shown in Fig. 1, a hierarchical structure was adopted in the CAMAC highway network corresponding to the hierarchical structure of the JT-60 control system described in the previous section. The ZENKEI computers of HIDIC-80Es are connected to the subsystems that constitute a major part of the JT-60 facility through the nodal crates called communication crates. Various types of CAMAC highways (central highways) are used depending on their data communication requirements [3]. Some of the central highways are composed of dual serial highways for reliability. As the NBI and RF heating systems and the data processing system for plasma diagnostic instruments have their own minicomputers and a general-purpose large computer, the ZENKEI computers are also connected to them through their dedicated inter-computer linkage buses. The ZENKEI discharge control computer is also connected to a front-end processor at the Japan Atomic Energy Research Institute (JAERI) computer center, where the JT-60 experimental database is created. For local data transfer, each subsystem controller has its own highways named subsystem highways, device highways and local test highways.

The module configuration of the communication crate on the central highways in the Ia and Ib computer systems is shown in Fig. 1. An auxiliary controller with D-port (ACD) is installed in the crate as a mail-box module for communication. A dual-serial U-port adapter is used for conversion of the signals from/to the serial highway to/from the ACD. Besides the ACD module, two intelligent modules are installed in the crate. One is a crate controller which is driven by the local test highway. The other is an auxiliary controller with a 16-bit microcomputer of Intel 8086 or HIDIC-08L (ACM). These were developed by Hitachi Ltd. and Toshiba Co. respectively. The crate controller, the ACD and the ACM can control the dataway in that order of priority.

The ACD is a 3-wide CAMAC module which interfaces to the backplane dataway and the serial highway D-port at the front panel. It has two 16-bit, 2 KW buffer memories and two 24-bit data registers where the replies in the handshake protocol of the communication are written and read.

As the ZENKEI computer system must supervise many subsystems, a more efficient method than CAMAC is required for the data communication between ZENKEI and the subsystem controllers. Hence, the protocol and data format of this transfer extends the CAMAC standard to include a so-called " variable word-length " transfer. A single command can follow a block of data in conjunction with the function of the dedicated serial driver for HIDIC-80E. This variable word-length data transfer can operate about 2.5 times faster than the standard protocol. Furthermore, since each subsystem was contracted to industry separately, the handshake protocol and data format for the communication between the ZENKEI computers and the subsystem CAMACs were standardized as JT-60 standards at JAERI.

These communication systems allow full data transfer in spite of the increase in the amount of data, in particular the increased amount of discharge and feedback control data in the JT-60 Upgrade.

5. DISCHARGE SEQUENCE CONTROL AND TIMING SYSTEM

5.1 Discharge Sequence

Discharge control involves establishing a set of condition parameters, implementing the discharge and collecting data resulting from the discharge. This process is repeatedly performed due to the pulse operation of the JT-60 tokamak. Various types of discharge sequence have been prepared for high power pulse discharge, discharge cleaning and test operation. The concept of the sequential control phase was introduced for synchronizing the status of ZENKEI and the controllers of the subsystems used as the discharge actuators. The standard shot interval was originally designed to be 10 minutes. The discharge sequence is implemented by command and reply messages between the Ib computer and the subsystem controllers in conjunction with trigger pulses for the actuators generated by the timing systems.

The functions of so-called "pre-shot check" and "post-shot check" on the machine status are prepared for the precaution of machine troubles. The Ib computer in association with the alarm systems installed in the Ia computer and the hard-wired interlock system performs the discharge termination for protection of the machines at the occurrence of abnormal events according to their levels of seriousness.

5.2 Timing System

In the design of the timing system, consideration was given to safety, precision and reproducibility, correction of disorder in control sequence and linkage with real-time plasma control. In order to satisfy these requirements, we made provision for the timing system to have (1) control interlock, (2) timing pulse transfer with response recognition, (3) event-oriented control and (4) command output by computer access.

The timing system is characterized by the following four kinds of CAMAC modules: a clock pulse generator (CPG), a timing pulse generator (TMG), a timing pulse transmitter (TGT) and a timing pulse receiver (TGR). The timing system is under the control of the Ib and Ib^R computers through CAMAC highways.

The CPG generates the 1-msec clock pulse which is used as an external count-down clock to the TMG. The clock pulse is also used as an interruption signal of the control cycle in the real-time plasma control. The TMG has five 16-bit presettable counters. Hence, the sequence control from 60 sec before the discharge to the termination of discharge is performed with an accuracy of 1 msec. Each counter can generate a timing pulse when the "AND" condition is satisfied with its time-up and the event signal given by the computer in the discharge sequence. Transmission of the timing signal is executed by a handshake procedure by using an "acknowledge (ACK)" signal in combination with the TGT and TGR modules. The TGT module can encode and transmit 16-channel timing signals and the TGR module receives and decodes them and replies the corresponding "ACK" signals to the TGT within 40 µsec. In case of faults occurred in the trigger pulse generation by TMG and the transmission by TGT and TGR, Look-At Me (LAM) signals are available for the Ib computer to terminate the discharge sequence.

6. PLASMA CONTROL

A control system has to be designed according to its control objective, which implies sufficient understanding of the system including the objective. Since this premise did not always hold for the tokamak plasma, the control system was required to have a wide range of flexibility to tune the control method. Reliability was also required from the view-point of availability of the control system. To satisfy these requirements, we decided to build a fully-digital control system with digital computers and CAMAC-standard equipments [4]. Thus, we were able to prepare many variations of plasma control structure, by changing the discharge condition setting. In spite of the system having this flexibility, we often changed the control system by replacing computers with more advanced ones.

In this section, the present system structure and performance for the JT-60U plasma control are described.

6.1 Outline of System Configuration

As shown in Fig. 2, the JT-60 plasma control system contains two feedback loops. The major loop controls plasma heating and gas fueling and the minor loop controls plasma position and current via five sets of poloidal field coils. The control cycle of each loop was determined by its control objectives. The cycle time of the major loop is 10 msec and the minor one is 0.25 and 0.5 msec.

In the major loop, the Ib^R computer supervises the controllers of the gas injectors and the NBI and RF heating systems by using a byte-serial highway for the transfer of status data and control commands during a shot. A dual-port memory module with D-port (DPMD), which has two 16-bit, 128-word buffer memories, is installed in each communication crate in the Ib^R highway as a mail-box module for the fast data transfer. In order to increase the data transfer speed, the byte-serial driver, which was developed for HIDIC-80E, has a "command buffer" to burst CAMAC functions and data to the highway.

A VME-bus system has the characteristics that (1) we can utilize advanced micro-processors with 32-bit or more accuracy, (2) its system clock (16 MHz) is faster than the CAMAC system clock (5 MHz), (3) VME-bus modules interfacing with CAMAC systems are on the market, etc. In the minor control loop, faster and more accurate computation were required for plasma position and current control at the JT-60 upgrade. Hence, we decided to supersede the old 16-bit minicomputer-based feedback control system (IIb) and the 16-bit microcomputer-based DDCs by VME-based systems equipped with advanced 32-bit microcomputers [5].

Table 1 Computer Specification

RISC Microprocessor [MVME181]
MC88100 RISC Microprocessor at 20 or 25 MHz
16 to 20 VAX MIPS
HIDIC-80E
Cycle Time: 0.48 μsec,
0.8 MIPS, 0.3 FLOPS

Table 2 Data Transfer Rate

HIDIC-80E (BSD) ~ DPMD	K words to 1 crate/module Time = 80 + 2.8 × K (μsec)
MVME181 (CBD) ~ ACB	K words to any crate/module Time = 2.4 × K (μsec)

The new IIb control system is composed of 4 VME racks, which are connected with each other through 6 bus repeater/expanders (PT-VME902A-1, Performance Technologies, Inc., U.S.A.). Three MC88100 based RISCs named MVME181 (Motorola Inc., U.S.A.) are installed in one of the racks. The other three racks are equipped with I/O modules. The parallel and pipe-lined processing of signal inputting, calculation of plasma state variables, feedback calculation and command output by using the three RISCs is synchronized by interruption from a 250-μsec clock pulse given by the timing system and the flags transferred between the RISCs through VME-bus. The programs in the RISCs are written in C language. The host computer for developing the VME microcomputer programs is a Sun3/140M workstation (Sun Microsystems, Inc., U.S.A.).

The old 16-bit microcomputers (i-8086 based ACMs) in the DDCs were superseded by VME-based 32-bit microcomputers named MVME147 (Motorola Inc., U.S.A.). I/O modules, however, are installed in the DDC CAMAC crates as before.

Another CAMAC crate is provided for transferring data between the IIb and the DDCs and between the IIb and the Ib computers. The VME racks in the IIb system are connected to the CAMAC crate via a CAMAC branch driver (CBD8210 made by Creative Electronic Systems, Switzerland), a branch highway and a type-A2 crate controller (CCA2 made by Standard Engineering Corp., U.S.A.).

Specifications on the computational speed of the computers used in the plasma control system are summarized in Table 1. Data transfer rates are also shown in Table 2.

6.2 Performance of Plasma Control

The newly developed plasma control system is working well in the initial experiments at the JT-60 upgrade. Stable deuterium plasmas with plasma current up to 5-MA and the NBI power of 20 MW have been produced. The plasma position and current are well controlled by PD (proportional and differential) control with matrix gain. For example, the differences of the state variables between the observed and desired values in a 4-MA divertor plasma are about 15 kA for the plasma current and less than about 1 cm for the plasma horizontal and vertical positions and the height of X-point (a cross point of the separatrix line of plasma boundary). Dynamic switching of the control scheme is available for sophisticated control in the initiation and termination of plasmas. The phase control also functions well for discharge termination in association with the computer and hard-wired discharge fault systems.

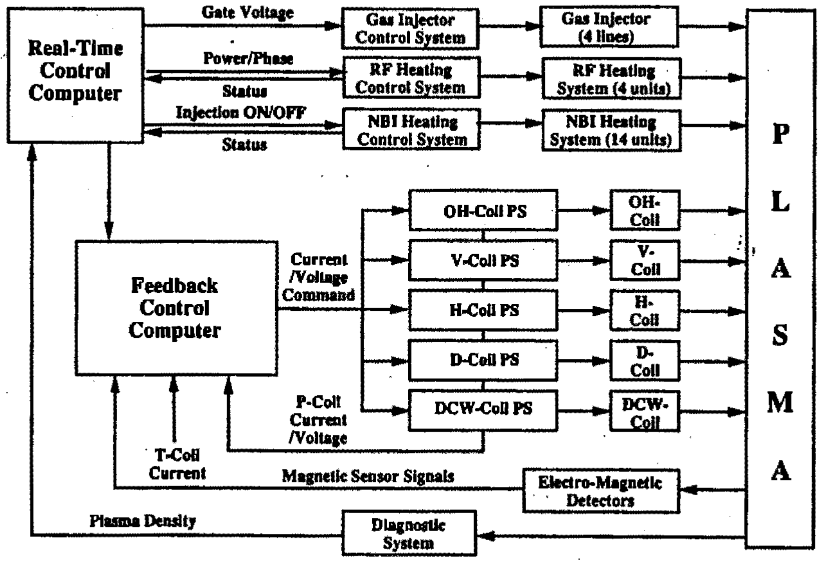


Fig. 2 Data flow of the JT-60U plasma control system

7. MAN/MACHINE INTERFACE

As the man/machine interface for the JT-60 plasma operation, the ZENKEI control system uses modern workstations for operator's consoles. A large mimic panel is provided for displaying overall status of the JT-60U. ZENKEI also has an ITV system and a broadcasting system. Each subsystem has its own console for the individual operation. The subsystem console is equipped with some CRTs having keyboards or light pens, button switches, mimic panels and alarm annunciators. These consoles are located in the JT-60 central control room. For individual operation of the subsysem devices, each subsystem also has its local console.

In the experience of the past five-year plasma operation, one of the most troublesome operations was the operation of setting the discharge condition parameters. As described in Section 5, this operation must be implemented within a short time during a shot-interval. Problems arose because of inefficient consistency checks among the condition parameters and lack of guidance for the setting. Improvement of the functions of plant monitoring and discharge results data display was also required for the efficient and comfortable operation in the JT-60 experiments. For example, the ZENKEI operator's consoles before the upgrade were equipped with old type color semi-graphic and B&W graphic terminals. We were not able to use Japanese characters in these terminals. Moreover, it was required that the layout of the consoles was changed in order to increase the number of consoles for the plasma diagnostic devices.

In order to cope with the above requirements, no more room to install new programs for the improvement was left in the original computer system because it was designed 10 years ago. Memory sizes were small. The peripherals were so old that we could not make "user-friendly" interfaces. Hence, the hardware which composed the operator's consoles has been superseded by the modern workstations connected through a network [6].

As shown in Fig. 1, the new man/machine interface at the central level consists of ten workstations: Sun3/80GX and AS4040 (SPARC) (Toshiba Co., Japan) as the operator's consoles, one workstation of Sun3/470GX (Sun Microsystems, Inc., U.S.A.) as a file server of the discharge condition and an 32-bit industry control computer of HIDIC-V90/45 (Hitachi, Ltd.) as a file server of plant monitoring and plasma control data (We shall call it SVP). The workstations are installed on ordinary OA desks. Hence, the layout of the consoles in the control room can be easily changed. The SVP computer, which works under a combination of real-time and UNIX operating systems, is connected to the workstations through an Ethernet network. The SVP computer can share the 4-MB memory with the existing minicomputers of HIDIC-80Es. The shared memory is used for transferring a large amount of discharge result data. A DMA-controlled 16-bit parallel bus, whose controller is named CLC-P, is also provided for transferring a small amount of data such as event data from alarms and the discharge sequence from the HIDIC-80Es to the SVP. The TCP/IP protocol is used for data communication on the network. The NFS (Network File System) protocol is also used as an application-level protocol for a large amount of data transfer. The application software in the new man/machine interface of these functions are written in C language. A window manager "SunView" is fully used in the displays of these control functions on the workstations. Multiple windows are displayed and Japanese characters are available. Click operation with a mouse can be used in the operation such as selection of items on the displays via "pop-up menus" and "panel windows" of "SunView".

The new ZENKEI man/machine interface has been fully used in the initial operation of the JT-60U. The load on the operators has been reduced. In particular, mistakes in the operation of setting the discharge conditions has been greatly reduced and the operational efficiency has been increased. This resulted in the decrease in the number of the operators. Some lessons remain from the view point of time response. For example, when the operator wants to display waveforms of discharge result data with a large amount of data points, he can not wait for the display without irritation.

ACKNOWLEDGEMENTS

The author wish to thank the members of Control Group in Division of JT-60 Facility I at JAERI for their collaboration in the development of the JT-60 control system.

REFERENCES

[1] H. Ninomiya, et al., "JT-60 Upgrade Device for Confinement and Steady State Studies," Plasma Devices and Operations, vol. 1, 1990, pp.43-65.
[2] I. Kondo, et al., "The JT-60 Central Control System," Fusion Engineering and Design, vol.5, 1987, pp.69-84.
[3] T. Kimura, et al., "Communication System in JT-60 Control," in Proceedings of the 10th Symposium on Fusion Engineering, Philadelphia, 1983, pp.398-404.
[4] K. Kurihara, et al., "JT-60 Plasma Control System," Fusion Engineering and Design, vol.11, 1990, pp.441-454.
[5] T. Kimura, et al., "VME Multiprocessor System for Plasma Control at the JT-60 Upgrade," IEEE Trans. on Nuclear Science, vol.35, 1989, pp.1554-1558.
[6] I. Yonekawa, et al., "A New Workstation Based Man/Machine Interface System for the JT-60 Upgrade," to be presented at this conference.

Conceptual Design of Centralized Control System for LHD

H.Kaneko,K.Yamazaki,Y.Taniguchi
National Institute for Fusion Science
Chikusa-ku, Nagoya 464-01, Japan

Abstract

A centralized control system for a fusion experimental machine is discussed. A configuration whereby a number of complete and uniform local systems are controlled by a central computer, a timer and an interlock system is appropriate for the control system of the Large Helical Device(LHD). A connection among local systems can be made by Ethernet, because a faster transmission of control data is processed by a specific system.

I. INTRODUCTION

The National Institute for Fusion Science(NIFS) was established by The Ministry of Education, Science and Culture in 1989 to integrate all of the inter-university collaboration in Japan for nuclear fusion research. The new and main project of Large Helical Device(LHD)[1] at NIFS was approved during the 1990 Japanese-fiscal-year(JFY). The LHD system presently under construction at the new site of NIFS in Toki-city, Gifu-prefecture, will be completed in the 1996JFY[2].

The LHD system, with superconducting coils, will be the first machine which can sustain the stationary magnetic field composing nested magnetic surfaces. A magnetic surface is generated from a combination of the toroidal magnetic field and the helical magnetic field. It is the outermost magnetic surface, which determines the confinement area. A ring plasma with a major radius 3.75m and an averaged radius of cross-section 0.65m is confined. A schematic view of LHD is given in Fig.1. The magnetic field is generated by a pair of superconducting helical coils and three pairs of superconducting poloidal coils. The stored magnetic energy in a typical 4T operation (the strength of the average toroidal magnetic field at the plasma center) is 1.63GJ. The initial plasma is usually produced by radio-frequency(RF) power at the electron-cyclotron-resonance(ECR). The plasma is subsequently heated by arbitrary use of 20MW neutral beam injection(NBI), 10MW ECR heating and 9MW ion-cyclotron-range-of-frequency(ICRF) heating.

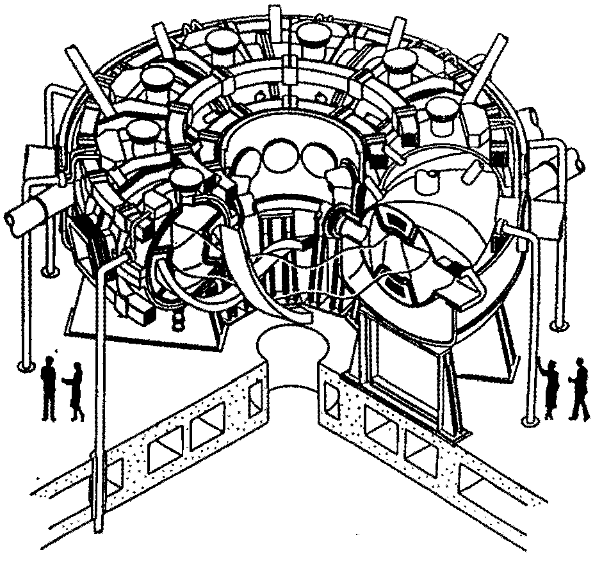


Fig.1 Schematic view of The Large Helical Device

The control system for LHD operation must be designed to ensure the safety of the whole LHD system which contains sensitive electronics as well as rough facilities handling large stored energy, high power, high voltage and high current within the same environment. It also has to manage an efficient performance of the plasma experiment. There are some distinctive characteristics in the control of such a large fusion machine still in an experimental phase.

The reliability is the most important factor in the present system. Since the power used is sufficient to cause fatal damage, the safety must be garenteed, including the case of an accident. A choice of reliable materials and an additional backup can ensure this reliability, but finally it is related to the available cost. Here, we consider only the logical reliability of the control system.

The flexibility of a configurational setup is very important for a machine in an experimental phase. Besides replacement, new types of facilities should be easily added to the system. Thus, the control system should be flexible.

There are various time scales to be controlled within. The real-time control in a pulsed plasma experiment is usually too fast to be treated by an operator's manipulation. Hence, the control of the plasma by an operator is made through pre-programming. Thus, the function of fast control necessary for plasma operation should be excluded from the role of a global controller. If necessary, such a function must be treated in a specific controller.

The pulsed characteristics of the plasma experiment, on the other hand, make the target very clear and simple. All facilities are operated in order to produce a high-temperature, high-density and well-confined plasma.

The last requirement is a possible local operation of some part of the facilities. When such an operation is demanded, global safety must be assured by the central controller.

Considering the characteristics of the LHD operation, we have discussed the conceptual design of the centralized control, which can more easily establish a logical relation between various manipulations. By the way, the data acquisition is closely related to the machine operation. However, the transfer of the vast plasma data from the data acquisition system to the control system is not necessary. Instead, a small amount of diagnostics signal will be directly provided to the local controller which needs the signal for the control. The diagnostics are loosely coupled to the central control system.

II. SUBSYSTEM

A. Functional Decentralization

All facilities contained in the LHD system have a local controller. The local controller is assumed to (1)fully control all of the devices under its control, (2)monitor the devices and the peripheral conditions, and (3)communicate with the central controller (also with the other local system in a restricted case) in order to establish mutual cooperation, as displayed in Fig.2. This unit is called a subsystem(SS). To simplify the whole control system the division into SSs must be optimized as to the quantity and quality of the communication, because the number of items to be communicated with others will be greatly different according to the division. An individual SS may have many sensors to assure the safety of itself. If a particular operation is not dangerous and has no necessity to cooperate with others, the operation can be managed locally. Such a inner-interlock, solved within the SS, gives the reliability and simple configuration. This decentralization, however, is designed to draw out more centralized control for the whole system.

B. Operation of Utilities

There are several SSs which are continuously operated to supply necessary utilities, such as, water, electricity, liquid helium and liquid nitrogen. Since the interaction of the facilities with the plasma experiment is infrequent, they could be operated independently. However, it also means that the load to the data transmission is small. The utilities can be operated within the same frame with the plasma experiment. This makes the whole control system uniform, and the system becomes transparent to the operator.

C. Plasma Production

There are two distinctive modes for plasma production. One is a pulse operation. The necessary electricity is stored in a fly-wheel generator and used in 10 seconds. The pulse can be repeated every 5 minutes. Although the amount of the control data is considerably large, they are managed before the start of the pulse by pre-programming. In designing the control system of the pulsed plasma machine, it is important to classify much of the control data to the pre-programming level, because they eventually determine the range of parameters adjustable by the operator. The real-time control of the plasma by the operator is thus excluded from the present design. When real-time control is truly necessary for some device, the function must be supported by a specific local machine.

III. CENTRALIZED CONTROLLER

A. Control Computer

When every SS is designed to be a complete local system, the role of a central controller is; (1)to establish the total safety, cooperation, collaboration and total efficiency of experiments and (2)to help an operator, who is usually monitoring some number of SSs at a time, by providing proper data, warning, guidance, and various routines useful in the operation. The centralized control of the whole system is mainly attained by a control computer(CC). Fast control for generating a proper time sequence for different SSs or an emergency stop due to an accident is one of the basic functions of the centralized system. This function should not be supported by CC but by an additional specific controller. When we use a timer system and an interlock system for this purpose, the configuration of the control system is displayed as in Fig.3, where the time scale of the action is distinguished as labeled on the right.

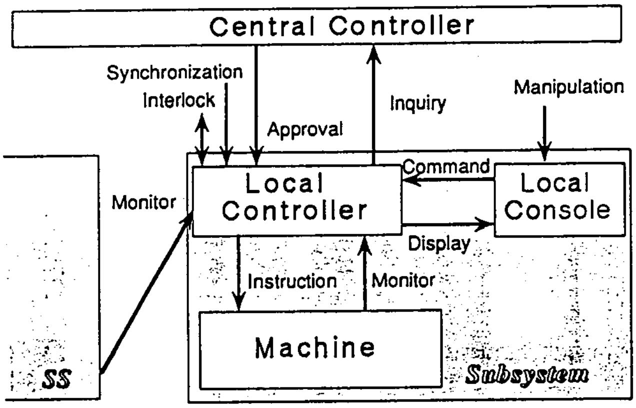


Fig.2 Flow of control data at a subsystem

B. Man-Machine Interface

As shown in Fig.2, every manipulation of a device by an operator is evaluated by the local controller, and then by CC when it is necessary, and the actual instruction is issued to the device from the local controller. In a centralized system it is not necessary for a SS to have its own console, because the command can be transmitted from CC. Since the information describing a manipulation is of the order of the manual input, the amount of transmission is never critical. Although a classical form of a direct input at SS is not excluded, control

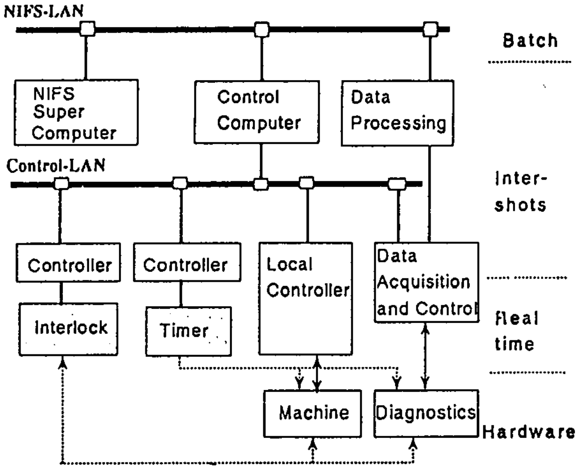


Fig.3 Structure of the control system

through the commonly-available general terminal of CC is considered to be the standard in the present system. Since a terminal can be used as a console for a number of SSs simultaneously, it is preferable for an efficient operation of the whole system by only a few operators. A number of workstations are used as a terminal of CC to provide a high-quality services as stated in (2) above.

C. Synchronization and Interlock

It is a basic function in a plasma machine to assure any SS to synchronize with the plasma production. This function is served by a specific timer. As already discussed, the control of the synchronization is completely pre-programmed. So the timer can be controlled by CC in the same way as a SS which is pre-programmable.

The other type of fast control is a safety interlock. In cases of emergency, the signal of the event must be transmitted much faster than the normal control data. The function must be also excluded from a design of CC. The signal is transmitted through a specific line to the device which needs a quick response. The use of this specific line must be restricted to the case where it is truly necessary. Since the action interrupt the uniform control of the whole system, it causes some vacancy of the control. A particular case will be studied later.

The reliability is sometimes insufficient in the circumstances with high power and high voltage. Then, a hardwired interlock is demanded for safety. The action is basically the same as the electronics. The action of the interlock system of both levels is monitored and recorded by a specific controller. The interlock system acts like one specific SS.

D. Network in Control System

A local area network(LAN) is used to connect the SSs and CC as schematically drawn in Fig.4(control-LAN), because an established standard of LAN is extremely useful to connect between different local controllers, each of which is differently designed from SS to SS by the person in charge. (It is not considered here to specify the local controlling computers in advance.) Also, it is inexpensive and flexible in the system configuration, and gives an easy method for electrical isolation, when an optical bus is used. To find a required network to LHD, the data transferred through the network is examined. There will be about 30 SSs. Although the amount of manipulation data is not so large, the amount of status data is considerably large in some of SSs, say 20kB. Considering the effective throughput, the rate of about 10Mbps is necessary. Then, the status data of 20kB correspond to 16ms for transmission. The operation of several such SSs at a time will be supported. Two possible standard of LAN are investigated: Ethernet and mini-MAP.

The standard Ethernet based on CSMA/CD seems eventually sufficient transmitting capacity for present application, if there is no problem on a use of optical fiber for the bus line. The adoption of Ethernet will contribute to the

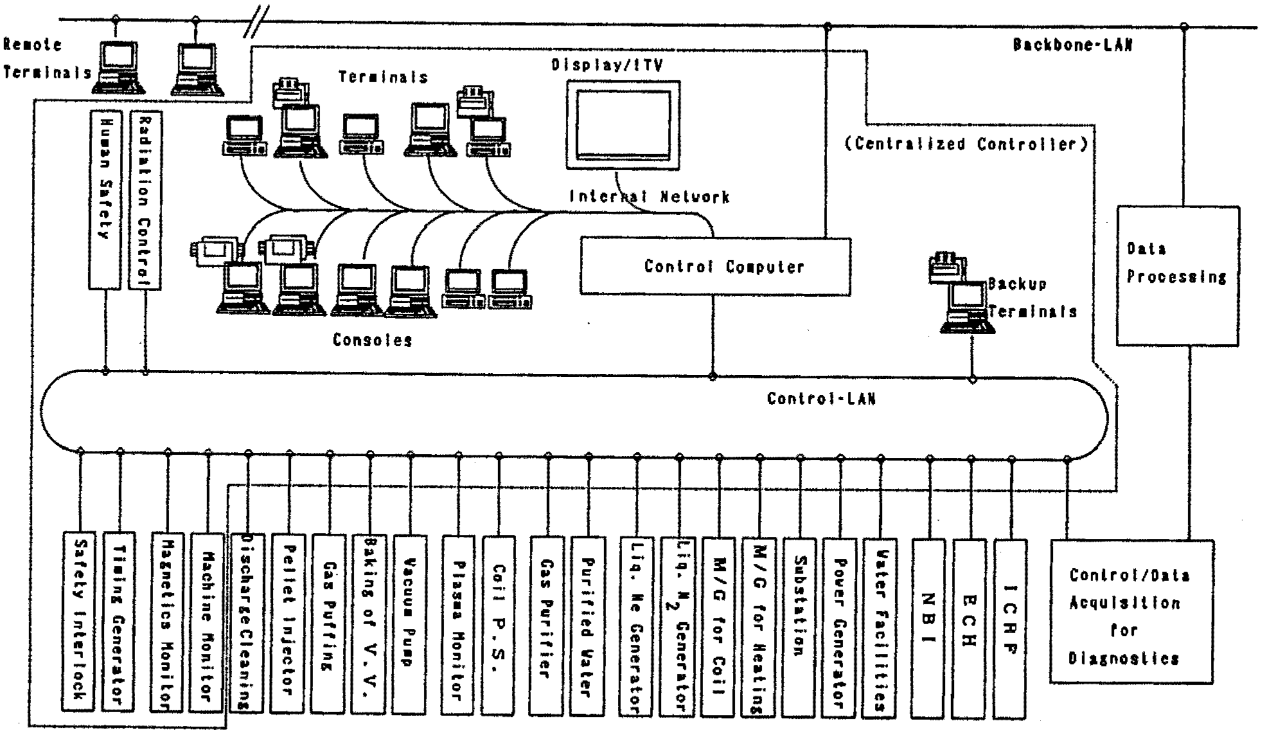


Fig.4 Global configuration of the control system for LHD

inexpensive construction of the centralized system. The use of fiber inserted into the coaxial bus is already proved to be effective to isolate between the divided segments. However, the insertion is rather expensive compared with the usual coaxial connection, because it does not contribute to the improvement of the transmission.

The mini-MAP based on token-passing seems more appropriate for the present application. The difference of the data to be transmitted, such as between the control command and the status report, can be treated by a distinctive service class in the token controller. A bus line by fiber is becoming popular. The use of mini-MAP is attractive for the design of the control system. However, the adoption might be difficult owing to the considerably high cost and some ambiguity in the standardization.

The design of control-LAN by Ethernet is selectively investigated.

E. Centralized System for LHD Operation

As already shown in Fig.4, the man-machine(man-controller) interface is basically connected to CC. A manipulation of SS is received by CC and then checked in reference to the current status of the whole system. An approved manipulation is analyzed into the command(s) to be transmitted to the concerning SS(s). The status report must be collected by CC in order to be served to the operator before he determines a proper manipulation at the console of CC. If once-a-second update of the status report for a few SSs is demanded for operation of each SS, the LAN might get choked. However, such a use of console is well processed, if the update is demanded only every 5 seconds, which will usually be sufficient compared with the manual response. The capacity necessary for the plasma control is not critical, because the control is pre-programmed. The sufficient capacity of CC will be utilized for allowing a flexible management of the control data for the plasma pulse. An experimental parameter, once selected and fixed for the coming shot, can be safely modified, because the centralized control can easily and quickly review the change. This function will be very helpful for a series of physical experiment.

The real-time control of stationary plasma is not taken into account in the present design. Such a control is possible only within the ability of the CC designed as above.

IV. EMERGENCY PATICULAR TO LHD

A quench of a superconducting coil is detected by many comprehensive monitors on the temperatures and voltages. The signal is transmitted to the interlock system. There are many SSs to be interrupted by the occurrence of a quench, when the quench takes place during the plasma production. However, such an emergency stop is not sufficient in the LHD system. There are many other requirements caused by a large stored energy; the current on a quenched coil must be decreased rapidly, say within 20 seconds; the induced voltage due to fast decrease of current must not exceed the insulation specification of the coil conductor; the production of runaway electron due to the induced voltage in the plasma vacuum vessel must be suppressed during the current reduction; and the transiently induced current due to mutual coupling must not yield an excess electromagnetic force. Therefore, gas puffing or some hard object insertion to the plasma vacuum vessel is applied for the induced voltage not to cause the production of runaway electron. Then, the power supply of the coils is switched to the emergency reduction mode. To avoid excess concentration of the current to a particular coil, the current on every coil including what has not quenched is reduced simultaneously. To setup a proper time sequence for those steps, the analysis of the quenching, transmission to other systems and synchronization are processed in the interlock system. After a satisfactory response to the quench there remains an emergency state in many facilities. The recovery is complicated and not predictable. However, a centralized system will be very effective to support the recovery.

A power cut is a trouble common to any control. Emergency stop of the whole sytem at the power cut is a usual action. However, there are some facilities which must not stop but rather be initiated in order to keep the whole system safe in the present case.

V. SUMMARY

A conceptual design of the control system for the LHD operation is discussed The fusion machine, with a large superconducting coil system and still in an experimental phase, requires an advanced control system for a reliable and efficient operation. To achieve a labor-saving operation and to minimized human errors there, it is concluded that the control needs a uniform structure over the entire system The CC is designed to be devoted in the logical cooperation among SSs, while every SS is designed to have a complete local controller for each. The manipulation by an operator is given preferably from a console connected to CC, because every manipulation should be checked and confirmed by CC. Then, the cooperation is managed by CC before the issue of the commands to local devices. The pre-programming for pulsed plasma control can be fully supported by the centralized CC system. However, an interlock system, which governs emergency action, is necessary to assure the safety in case that an accidental event has occurred. Thus, fast transmission of emergency data is differently processed from the normal control data. Then, the use of Ethernet for connection among CC and SSs is an inexpensive choice for the present requirement.

Since the relation between SSs must be definitely controlled for the safety of the whole system, a completely decentralized control is impossible. The SS is completely designed to allow CC more centralized. The CC can be regarded as a specific subsystem taking only the role of centralization.

VI. REFERENCES

[1] A.Iiyoshi, et al., Fusion Tech. 17(1990)169.
[2] K.Yamazaki, et al., this Conference.

STATUS OF LHD CONTROL SYSTEM DESIGN

K. Yamazaki, H. Kaneko, Y. Taniguchi, O. Motojima and LHD Design Group
National Institute for Fusion Science,
Chikusa-ku, Nagoya 464-01, Japan

Abstract

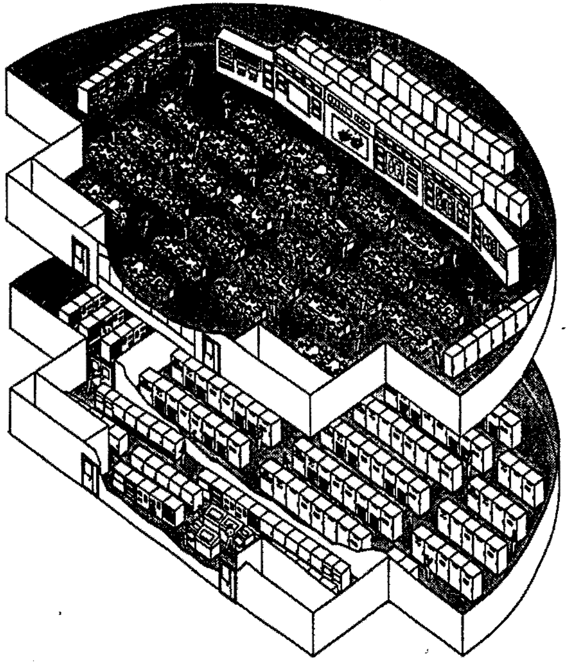
The present status of LHD (Large Helical Device) control system design is described, emphasizing on the plasma operation modes, the architecture of the LHD control system, the real-time plasma feedback system with PID or Fuzzy controllers and the construction schedule of the LHD control system. The conceptual and detailed designs are under way taking flexible and reliable operations for physics experiments into account.

I. INTRODUCTION

The Large Helical Device (LHD) fusion system [1-3] using 1.6 GJ superconducting (SC) magnet is now under construction and its plasma experiments will be started in April, 1997. For this purpose, a new national institute (National Institute for Fusion Science) was established in May,1989, and a new site (Toki city; one-hour drive from the present site in Nagoya) were prepared for these experiments. The main objectives of the LHD project are

(1) the study of the behavior of high temperature / high density plasmas using helical torus device for comprehensive understanding of toroidal plasmas, and
(2) the exploration of the prospect to the steady-state helical system reactor.

The major plasma radius of LHD is 3.9 m , and the magnetic field strength is 3 Tesla (4 Tesla in the second experimental phase), which is the largest SC fusion machine now under construction. To keep flexible and reliable operations of this SC machine, a new control concept is required.



In this paper, the present status of the control system for operations and experiments of the LHD system is presented.

II. LHD MACHINE DESIGN AND CONTROL CONCEPT

The LHD system consists of one pair of SC helical coils, three pairs of SC poloidal coils, plasma vacuum vessel, cryostat, vacuum pumping system, electric power supplies, plasma production system, liquid helium refrigerator, three (NBI, ECH, ICRF) plasma heating systems, many plasma diagnostic systems and so on. All these equipments should be monitored and controlled mainly from the LHD Control Building. Especially, the control system should be flexible as a experimental machine and reliable as a large plant.

In contrast to present helical devices, the LHD is characterized by the steady-state operation using the superconducting helical coils and the built-in divertor, which requires the elaborate control scheme for operational safety and the new plasma feedback system for experimental flexibility.

These LHD machine and central control systems are schematically shown in Fig.1.

III. LHD OPERATION SCENARIOS

The LHD machine operation is divided into three modes; all shut-down mode, facility operation mode and experiment mode. The experiment mode consists of the SC magnet operation mode and the plasma experiment mode (Fig.2). These modes are defined for clarifying the personnel entrance permission, magnetic field hazard and possible radiation exposure. Aside from software interlocks, the hardware interlock logic should be determined independent of these modes.

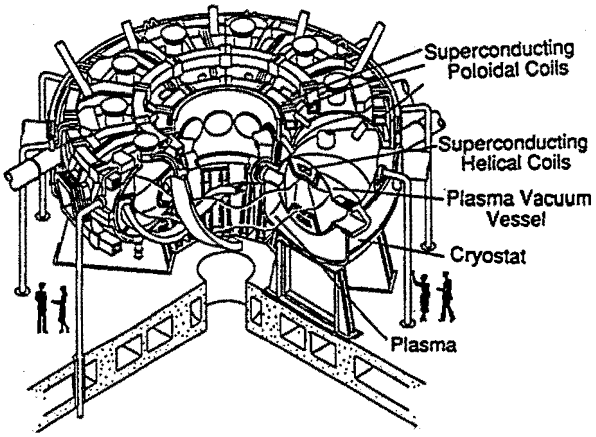


Figure 1 Schematicdrawings of LHD machine (right) and its control system (left).

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S05SRN08

The SC magnet will be operated for about 10 hours per day, and the number of short-pulsed plasma operations will be typically 50 - 100 shots per day. After 3-year first-stage experiments, the magnetic field strength is upgraded from 3 Tesla to 4 Tesla, and thousands of D-beam / D-plasma operations are planned.

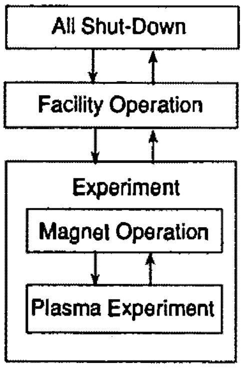


Figure 2 LHD operation mode

IV. LHD CENTRAL CONTROL SYSTEM DESIGN

Based on the above-stated operation scenarios, the designed control system is composed of central experimental control system, and several sub-supervisory control systems such as torus machine control, heating machine control, diagnostic control and electric / cooling utility control systems, as shown in Fig.3. All sub-supervisory systems are connected by the local area network (LAN).

Within the facility operation mode, basically almost all equipments are operated by each subsystem controller. The vacuum pumping and wall conditioning including baking and glow discharge cleanings are controlled from the main torus control system, and each heating system is operated by each controller. On the other hand, in the experiment mode, main input parameters are controlled mainly from the engineering workstation of the central experimental control system. Overall system diagram of LHD control systems is given in Fig. 4.

The details of this system design is described in the separate paper [4].

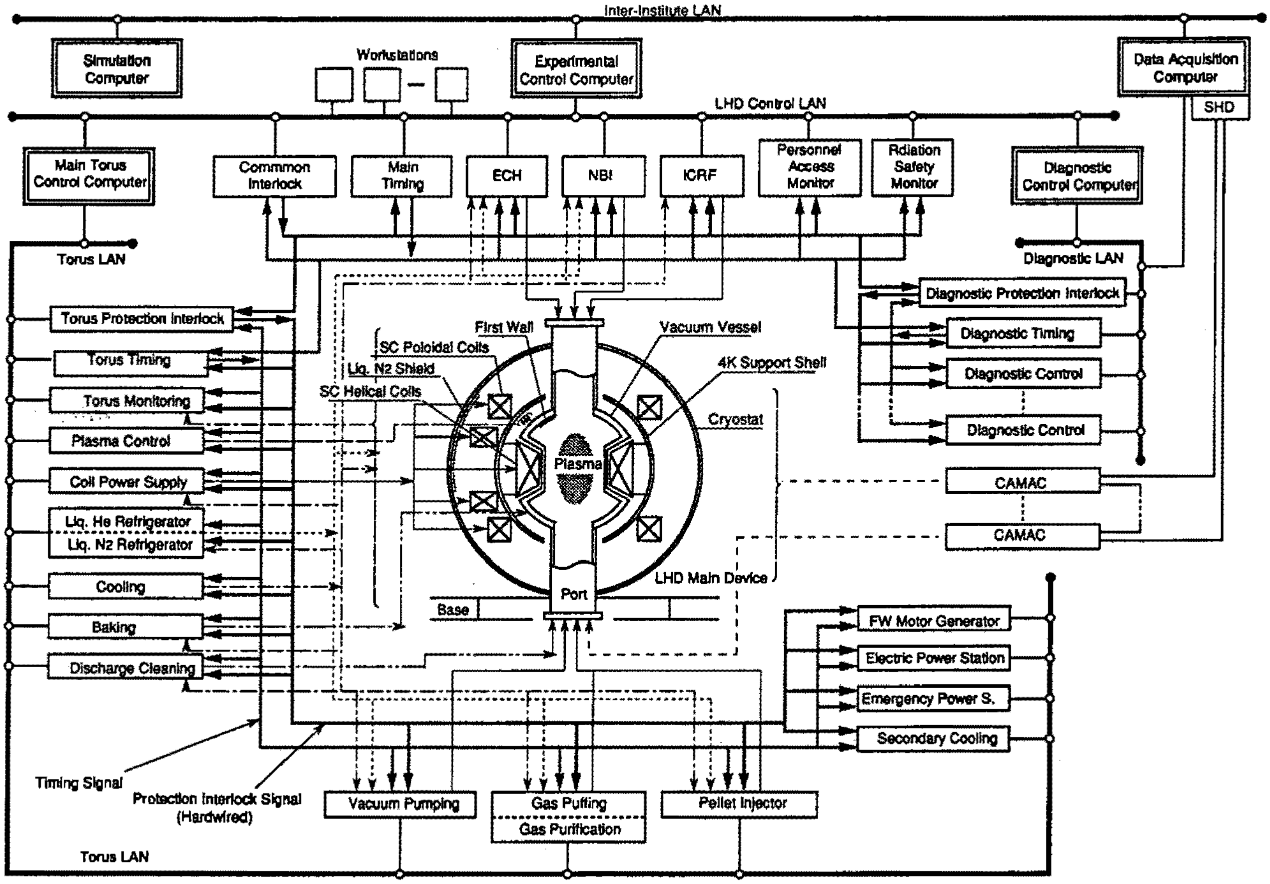


Figure 4 System diagram of LHD control.

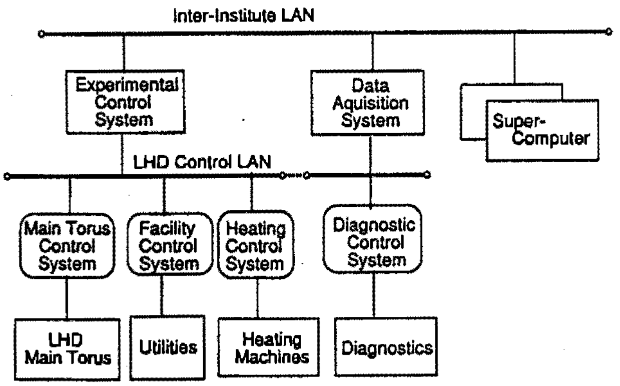


Figure 3 LHD control system architecture

V. REAL-TIME PLASMA FEEDBACK SYSTEM DESIGN

According to the experimental requirements to make flexible plasma operations, an elaborate feedback control system with PID and Fussy logic control concepts [5,6] are under consideration.

The quantities for feedback are plasma current (Ip), plasma position (Δ), plasma cross-sectional shape(κ), plasma density, heating power and so on. Especially, the former three variables are controlled by the power supplies of one-pair three-block helical coils (HF) and three-pair poloidal coils (OV,IS,IV). The basic concept of the plasma feedback system is shown in Fig.5 as a combination of the coil current feedback (IHF,IOV, IIS, IIV), the vacuum magnetic field feedback (B0,BD, BQ, ϕ) and the plasma feedback. These components are related to each other trough the magnetic configuration matrix. A typical system diagram of this scheme is shown in Figs.6 and 7(a).

Conventional PID controllers or min-max Fuzzy logic controllers with the center of gravity defuzzification method are analyzed including eddy current loops of the LHD system. The controllability of these two algorithms is compared in Fig.7(b) for the LHD Plasma operations. The details of this analysis will be published somewhere.

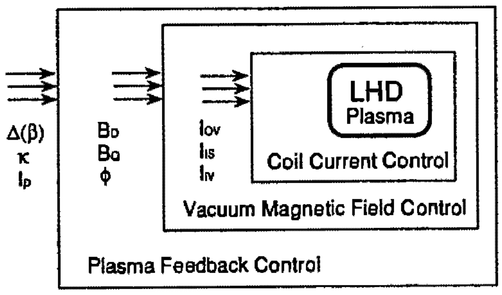


Figure 5 LHD plasma control concept

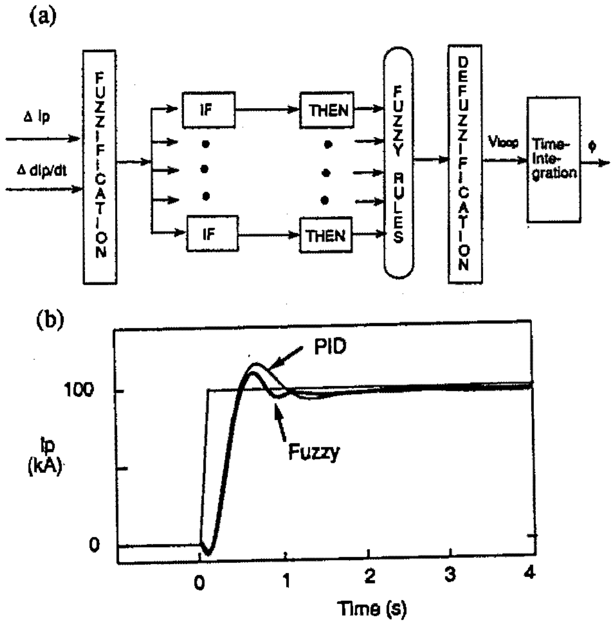


Figure.7 Fuzzy logic control system
(a) Fuzzy algorithm
(b) Comparisons between PID and Fuzzy feedback responses.

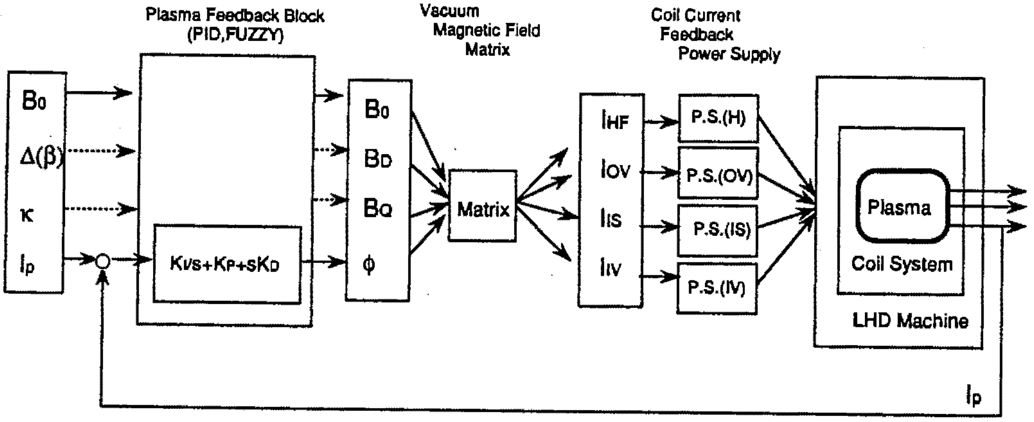


Figure.6 LHD Plasma control system diagram with PID controller.

VI. CONSTRUCTION SCHEDULE OF CONTROL SYSTEM AND BUILDINGS

The construction of LHD device itself has been started in 1991 as a 7-year project, and a first plasma operation is scheduled in 1997.

The basic design of the LHD central control system has been carried out in 1991-1992, and the detailed design will be done in 1993-1994. The proto-type R&D system for LHD machine operations and plasma controls is under preparation in 1991-1996 for the development of software. The LHD central control system will be constructed in 1995-1996.

In the new site the cryogenic building was firstly constructed in 1990 and the main LHD building is now under construction and will be completed in 1993. The LHD control building will be completed in F.Y.1995, and the central control devices will be installed there.

VII. SUMMARY

The design of the control system for the Large Helical Device (LHD) system has been conducted for the flexible and reliable operations of the large experimental fusion systems. The LHD machine and its control system will be completed in March, 1997.

ACKNOWLEDGEMENTS

The authors thank Dr. S.Hidekuma, S.Kubo, S.Okamura, S.Tanahashi, M.Fujiwara and engineers of Omika Works, Hitachi, Ltd. for stimulating discussions.

REFERENCES

[1] A. Iiyoshi, et al., Fusion Tech. 17(1991)169-187.
[2] O.Motojima, et al., Plasma Physics and Controlled Nuclear Fusion Research, IAEA-CN-53/G-1-5(1990)
[3] K.Yamazaki, et al., Plasma Physics and Controlled Nuclear Fusion Research, IAEA-CN-53/C-4-11(1990)
[4] H.Kaneko, K.Yamazaki, Y.Taniguchi, this Conference.
[5] L.A.Zadeh, Inform.Control, 8(1965)338.
[6] E.H.Mamdani, Proc.IEE,121(1974)1585-1588.

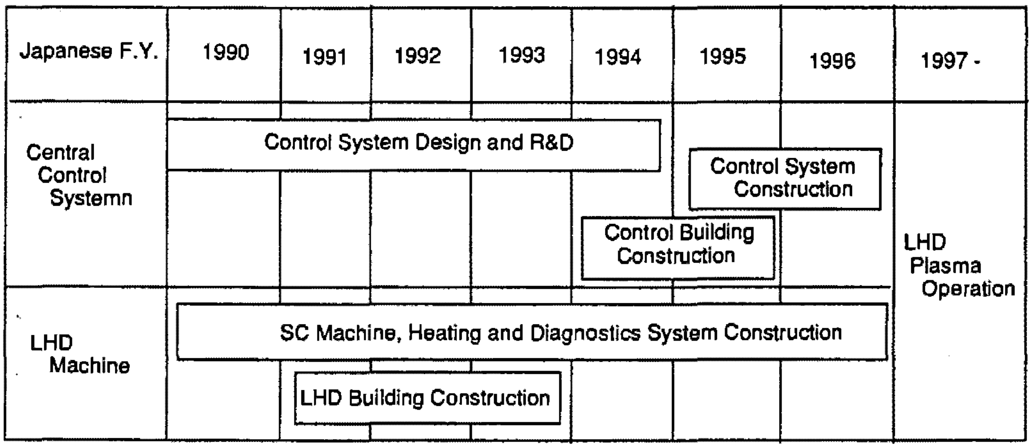


Figure 8 Construction schedule of LHD machine and central control systems

Control and Monitoring System Design Study for the UNK Experimental Setups

A. Ekimov, Yu. Ermolin, M. Matveev,
S. Ovcharov, V. Petrov, V. Vaniev.
Institute for High Energy Physics,
USSR 142284 Moscow region Protvino P.O.Box 35

Abstract

At present a number of experimental setups for the new UNK project are under construction. A common approach to the architecture of control/DAQ/trigger systems will be used in the development of electronics for all these detectors. The system analysis and design group has been formed for this purpose. The group activity is aimed at the development of such unified system. The group has started with control and monitoring system as one of the most important parts and the environment for the DAQ/trigger systems. The group activity status report is presented.

1. INTRODUCTION

The construction of several experimental setups is planned in the frame of the new UNK project at the Institute for High Energy Physics (Protvino) [1-4]. The size and complexity of these detectors is by an order of magnitude bigger as compared with those at the existing proton synchrotron. Table 1 shows the number of channels, event size and readout event rate for old and new (proposal) experiments.

TABLE 1

Exper.	Number of ch. *10^3	Event size (kbyte)	Event rate (Hz)
FODS-2	1	0.1	100
PROZA	1.5	0.5	250
SPHINX	25	1.5	100
VES	13	1	1000
NEPTUN	100	25-50	1000
GLUON	13	3	500
MPS	80	25-100	600
MMS	60	15-20	50-100

The general requirements to the electronics for the detectors have been formulated in the technical proposals. The electronics will be developed both at IHEP and at other institutes and organizations who take part in the detector preparation. The use of industry electronics is also foreseen. Such a distributed method of the development requires management and technical coordination when designing the electronics. A working group for the preliminary study of the design approaches and electronics architecture has been formed at IHEP.

2. DESIGN APPROACH

At the beginning it was decided to choose the approach for the design of the electronics for different experimental setups, namely: to use specialized electronics designed for the specific experimental setups, or to use unified electronics adapted for a specific experimental setup. Both of them have well-known advantages and disadvantages. The choice is defined by the size and complexity of the detector, available funding and manpower, time schedule and so on. For the experimental setups at UNK it is possible to note the following points:

- experimental setups for the UNK are constructed practically at the same time,
- IHEP will participate in the design of the electronics for several (or all) experiments,
- the preliminary analysis of the electronics requirements for different experiments shows the existence of common functional elements.

With an account of the experience in using unified electronics for the experimental setups at the existing accelerator it seems reasonable to use the second approach for the electronics design at UNK.

As a second step the number of points to be studied at this preliminary stage have been defined with the aim to make some recommendation for the following investigations:

- standard modular unified architectures of the electronics for the experimental setups divided by the numbers of functional subsystems and levels with the definition of an unified interfaces,
- more detailed definition of this general architecture with the standard implementation of some parts of the system and possibilities of adapting these parts to the specific demands of the experimental setup,
- standard framework of the electronics development (tools, test setups management and so on),
- existing experience at IHEP in these areas.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S05SRN09

3. ARCHITECTURE

3.1 General

The discussion around the general architecture is based on the analysis of the technical proposals of the experimental setups and the outside experience in the field [5-8]. As a result it has been decided that all electronics can be split in several functionally independent parts, interacting during the operation:

- data acquisition system for the hierarchical data collection of the useful detector information,
- multi-level trigger system for the event rate reduction and useful event tagging,
- technological equipment of the experimental setup, such as high voltage, gas and power distribution, calibration and so on,
- control and monitoring system, which must provide supervision of all infrastructures of the experiment.

Such a separation with the definition of interfaces gives the possibilities of more or less further independent upgrading and modification of the electronics without redesigning the whole system.

The control and monitoring system of the experimental setup from this point of view is that frame where all the other functional subsystems are working. During the experiment run it should provide the status information and the remote access to all parts of the electronics. During the experiment preparation it may be used as a setup for the electronics debugging and testing. Due to this development of the control and monitoring system could be considered as a first step during the electronics design.

For better understanding of the control requirements for other parts of electronics some preliminary considerations of their structure and possible implementation have been made.

3.2 Data acquisition system

The architecture of the data acquisition system should allow us to easily scale and adapt to the experiment requirement, independent development of separate parts and further smooth upgrading. Natural parallelism of the event data and pipeline structure for high performance should be exploited. Proceeding from above supposition we propose to use three functional level data acquisition system:

- front end electronic,
- readout and event building,
- event processing and data storage.

The internal structure of subdetectors specialized front end electronics includes a number of basic components (analogue storage, converters, digital memory). The packing of the front end electronics depends on the technological possibilities at the moment of the system realization. At present we assume the crate implementation of the front end.

Possible implementation of the front end channels readout and event building electronics has several directions (shared bus, dual port memories, work on SCI). Dual port memory realization of the event building seems the most suitable for us due to the possibilities of realization and some experience we have. The hardware of this part of data acquisition is shared between front end and event processing crates connected by the cables.

Event processing will be done by the farm of commercial single board computers housed in the crates. Each node has input buffer, processor and output buffer, and processes all event data without passing them to other processors.

3.3 Multi-level trigger system

Event selection will be performed in three stages (or levels). Rough decision at the first level allow further conversion of the event data. A more precise second level decision tags the event for the readout. The third level performs full event processing (as a part of data acquisition).

Without going into details of different trigger processors implementation it is possible to note some control and monitoring requirements. In the case of analogue first level trigger processor implementation the control system should provide careful calibration and has no influence on the processor during the operation. For the second level processors (mostly digital), as for the data acquisition electronics, access to all internal registers for the test purpose seems essential.

3.4 Technological equipment

The general infrastructure of the experimental setup consists of several different technological subsystems. For the control and monitoring of such equipment specific electronics will be designed. It should give the possibilities for the access from the control system even in case of non-standard implementation.

4. CONTROL AND MONITORING SYSTEM

Following the above functional subsystems partitioning of the electronics, control and monitoring system may also be divided into the control nodes. The task for this control node is supervision of the allocated part of the electronics. It should provide the interface to the user of this subsystem and the interface to the whole system. Each of the nodes consists of three main parts:

- local computing power,
- interface to the controlled equipment,
- LAN between control nodes and the system.

A local computer may be a personal computer or a single board computer, housed in a standard crate. It keeps all the information about allocated electronics (configuration, values of all parameters and so on). At present two possible implementations of such a local computer are under estimation: the AMS like crate with I8086 processor and VME crate with Motorola 68000 processor.

Interface to the equipment is defined by the implementation of the specific part of the electronics. Because of the lack of any final decision made about housing of different parts of the electronics, only some preliminary consideration may be fixed. We assume the crate implementation of the electronics. Inside the crate the control path may be physically separated from or joint with the data acquisition or trigger data path. Each crate for the control node is a station on the inter crate bus with the multi-register access and geographical addressing possibilities. The controlled crates are described by uniform table, accessible for the read/write operation.

For the communication between local nodes we plan to use commercial equipment like Ethernet or Token Ring.

Some experience of the development of such a separated control and monitoring system has gained during the development of the data acquisition and control system for the Big liquid ARgon Spectrometer (BARS) of the Tagged Neutrino Experiment (TNE) at IHEP. It consists of:

- I8086 based microcomputer housed in a CAMAC crate (analog of a control node), works under specialized operation system,
- interface to the controlled equipment, housed in a CAMAC crate, implemented as an auxiliary controller,
- star interconnection network to the main control computer (IBM PC).

The task for this control system is calibration of QDC cards, testing of digital modules and control of the data acquisition and triggers module during the experiment run. In a reduced configuration such a system is used for debugging and testing the electronics modules.

Several prototypes of special equipment for the user interface have been developed at IHEP.

ACKNOWLEDGEMENT

The authors would like to thank Doctors A. Sytin and V. Senko for the task definition and Yu. Simonov for the fruitful discussion.

REFERENCES

[1] A. M. T. Lin, Spectrometer Design for High Energy Spin Physics at UNK, Proc. of a Workshop on Science at the Kaon Factory, Vancouver, BC, Canada, July 23-28, 1990, vol.2, sect.6, editor D.R.Gill.

[2] Yu. D. Prokoshkin, The Experiment to Study the Gluon Interactions and Glueball Production in the Central Region of Hadron Collisions at UNK Energies (2-3 TeV), IHEP Preprint 85-32, Serpukhov, 1985.

[3] A. M. Zaitsev, Status of the MultiParticle Spectrometer (MPS) Facility, Proc. of Workshop "Physics at UNK", Protvino, 20-24 March, 1989, p.27.

[4] V. V. Abramov et al., Study of Multimuon Events in 0.4x3.0 TeV Proton-Proton Collisions with Multimuon Spectrometer MMS, (Proposal of experiment on UNK collider), IHEP Preprint 90-93, Protvino, 1990.

[5] N. Ellis, S. Cittolin, L. Mapelli, Signal Processing, Triggering and Data Acquisition, Proc. of Large Hadron Collider Workshop, Aachen, 4-9 October 1990, CERN 90-10, ECFA 90-133, vol.1, pp.505-538, Editors: G.Jarlskog, D.Rein.

[6] A. J. Lankford, Computing and Data Handling Requirements for SSC and LHC Experiments, Computing in High Energy Physics, Santa Fe, 9 April 1990.

[7] M. F. Shea et al., The DZero Downloading and Control system, Proc. Europhysics Conference on Control System for Experimental Physics, editor B.Kuiper, CERN 90-08, p.84.
R. Goodwin et al., Initial Operation and Current Status of the Fermilab D0 VME-Based Hardware Control and Monitor System, Nucl. Instrum. Methods A293(1990)125.

[8] A. K. Amundsen et al., The Control System of the OPAL Detector at LEP, Nucl. Instrum. Methods A293(1990)145.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S05SRN10

Hard- and Software for Measurement and Control of the Pulse Thermonuclear Installation

V.V.Bulan, E.I.Dudorova, V.I.Zaitsev, A.V.Kartashev,
Yu.N.Lusin, A.I.Nebogin, G.M.Oleinik, Yu.V.Papazyan,
V.N.Savochkin, V.P.Smirnov, V.M.Chikovsky
Branch of I.V.Kurchatov Institute of Atomic Energy,
Troitsk, 142092, USSR

Abstract

This paper describes control and measuring systems of the pulse thermonuclear installation "Angara-5". The "Angara-5" operates in a monopulse mode. It takes a long time to prepare the installation to the work shot . The main information flow about the installation output parameters and the target processes comes for 10^{-7}-10^{-8} sec. The measuring-control equipment has a multi-level hierarchy structure where the lower level is local systems controlled by own computers. Measuring systems contain waveform digitizers different types. The supervisor console system realizes the communications with the local systems, as well as the data acquisition, processing and storage. Hardware and software structures are given. Careful equipment shielding and grounding have provided level of noise 30 mV. Fast signals processing features are discussed.

I. INTRODUCTION

High power pulse generators had been using at first as accelerators of high current electron beams [1], lately have founded use as a driver technology for inertial confinement fusion experiments. Such generator can produce electromagnetic pulses with power 10^{13} - 10^{14} W and duration < 10^{-7}sec on load. Type of load is determined by experimentally program taken place at the installation. High voltage diodes producing intense ion beams are used as load at some experiments. Beams energy has been transporting on thermonuclear target [2]. Gas jets or liners different types are used as load at other experiments. In this case ions of load are accelerated to axis by means magnetic field of current through load [3]. It's necessary to note next features of such installations that determine structure of control and measuring systems:

-small duration of processes in installation after start (~10^{-6} sec). It excepts possibility of control at regime "on line" and requires application of fast analog-to-digital converters with buffer memory,

-seldom work starts of installation (few starts in day) make easier requires to systems of before-starting preparation and to data processing rate after shot. It's allows to design systems on base interfaces like CAMAC,

-high risetime of currents and voltages (~10^{14} V/sec, A/sec) provoke high level electromagnetic noise and requires the special design on electromagnetic compatibility of equipment.

Such installations are specific systems and requires design the special hard-software for effective working.

This paper describes the realization experience of control and measuring systems on the pulse thermonuclear installation "ANGARA-5" [4]. The installation consists of 8 modules worked synchronously on common load. Parameters of installation ($U=1.5$MV, $I=4$MA, $T=10^{-7}$ sec) allow to provide a different experiments on thermonuclear targets heating.

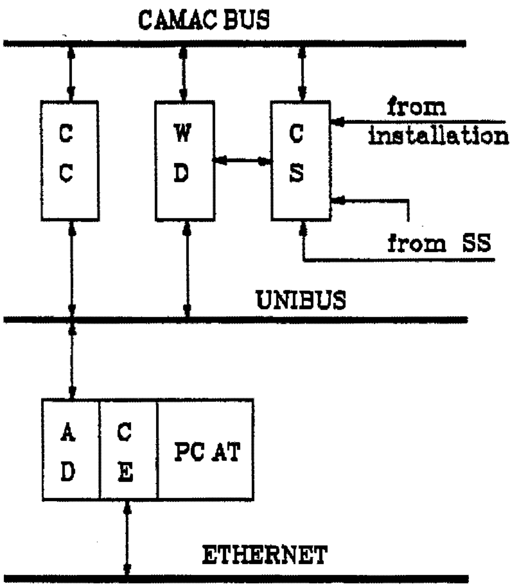
II. HARDWARE

Hardware consists of separate local systems and has multilevel structure. Local system fragment is shown in fig.1. The CAMAC crate blocks or their VECTOR [5] analogs are placed at the low level and are controlled by means of CC. The CAMAC crates are connected in systems by help UNIBUS and controlled by means of PC computer through adapter. Local systems are united in complex and connected with host computer center by ETHERNET.

The technological parameters system ST (realizes control before-starting preparation) and the synchronization system SS (determines moments of switching on installation parts and measuring systems) are control systems in the usual sense. Let's consider of structure (fig.2) and functions of local systems.

Supervisor system SD produces local systems control and provides date acquisition, archiving, processing and display imaging of information from all local systems.

Technological preparation system ST realizes all necessary operations before installation shot. A main operations are measurement of slow changing parameters (distilled water resistance, gases pressure in switches, voltage on condensers etc.) and control of gases pressure and condenser charges processes.



CC : crate controller
WD : waveform digitizer
CS : commutator
AD : adapter PC–UNIBUS
CE : controller ethernet

Figure 1. Typical schema of local system.

It's possible to measure technological parameters on 256 channels and to regulate pressure in 128 volumes. Technological parameters slow change allows to use CAMAC blocks commercial set.

Synchronization system SS organizes work of all measuring systems in the same time scales "attached" to installation work cycle. The set of VECTOR blocks allows to generate starting signals at different moments of cycle. Time scales is determined by 100 MHz frequency generator. Timeseting blocks "attachs" synchronization pulses to cycle with accuracy 2,5 nsec. With purpose of universality the system is equipped of blocks for multiplication and amplification of synchronization signals.

Fast processes measuring are produced by systems SW1,SW2,SG. These systems contains fast analog-to-digital converters different types and includes both commercial CAMAC or VECTOR blocks and special developed devices. It's necessary to have the information about parameters of 200 signals in each shot . Main installation processes have the duration 10^{-6}-10^{-8} sec and contain ~100 Kbyte (without video image information). The systems architecture are designed according to types of devices. Signals sorting according to users requires is provided on software level in the supervisor system.

System SW1 contains 32 scale-time digitizers UPN-92 producing signal waveform measuring with time step 1 nsec. Fragment of SW1 system is shown in fig.3.

System SW2 includes 32 measuring channels with a different types of real-time waveform digitizers. Measuring step on time is from 10 nsec to 50 nsec.

Characteristics of digitizers basis types are shown in table 1.

Table 1

Characteristics of digitizers basis types

Types	Bits	Time step	Steps number
UPN-92	8	1 nsec	256
BPN-93	7	15 nsec	128
F-4226	8	50 nsec	1024

System SG contains less expensive digital devices for pulse general parameters measuring and includes 64 channels of pulse start time measuring, 64 channels of gated time integrals measuring and 16 channels of peak amplitudes measuring.

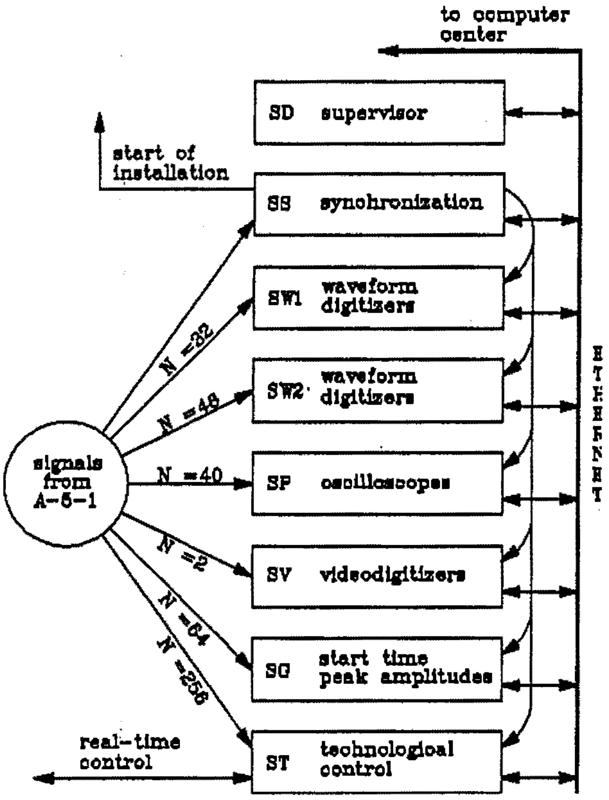


Figure 2. Hardware composition.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S05SRN10

Video image processing system SV is intended for reception of data arrays about load radiation image. There are used as coding devices:

- mechanical density measurement device (AMD) scanning image on photographic film (step = 5 Mkm) and transferring blacken density to digital code,

- image digitizer on base of charge transfer devices.These devices application as positional sensible detectors allows to receive information about allocation of intensity of plasma radiation in different range of waves lengths (X-ray too). In this case distribution of intensity are registered without film processing.

Physical diagnostics system SP serves of devices complex intended for registration of load radiation (neutron, optical, X-ray and etc.) and includes oscilloscopes different types. Highest band width is 1.5 GHz. System functions are preparatory operations (service of cameras, oscilloscopes, generation of a test signals), registration and processing of signals from neutron activating analysis detectors. Commercial CAMAC blocks are used as control hardware.

All pulse measurements systems contains necessary equipment for automatic amplitude-time calibration. Vertical deflection factors, time base sweep curves and time start reaction are determined and stored on disk for each digitizer. The special electrical supple, careful shielding and grounding of all measuring and control systems, galvanic isolation by optic devices are used for electromagnetic compatibility. As result, a peak-to peak noise level of 30 mV has been measured from cables located in the noisiest region [6].

Figure 3. Fragment of SW1 system.

III.SOFTWARE.

The hardware specific and installation ANGARA-5 peculiarities have required specific software design. All software blocks are "canned" for users. Special menus provide users interaction with hard-software. Main blocks of software structure is shown in fig.4.

Dispatcher program occupies the central place in the structure. Program supports next tasks:

-data reading of experiment description program (DSR), preparation them for transmission to local systems,

-definition of equipment work regime (calibration, work shot),

-initialization of connection with local systems, preparation of local systems to work,

-start of the synchronization system, data reading from local systems and transmission them to data base,

-graphic data presentation.

Hardware adaptation to concrete experiments is executed by means of DSR program. The program provides:

-holding and editing of installation sensors parameters,

-holding and editing of synchronization channels parameters,

description of signals transmission lines including coefficient of signal fading in cable,

-schema of registrars switching.

The information is transmitted from DSR to signals file of data base by means of D-program in each shot.

Object specificity have required development of the object-oriented data base DB with program interface and developing possibilities of graphic presentation. Data base knows 3 type of object: signals, amplitudes, times. Every signal saved in DB consist of fixed and variable parts. Fixed part contains ancillary information about sensor type, measuring schema etc. Variable part contains signal itself. Length of the part depends of digitizer type.

Manager of data base allows to look through data base on different indication, to extract and edit a data.

Local systems during exploitation are the most evolution part in respect of its composition, because necessity in substitution of coding device type arises constantly. That's why programs LS was built so, that they could be easy adapted to change of there compositionally. Local system is divided on subsystems for which was built files of schemas description. External file describes connections between subsystems. One can make out next elements in software structure of local systems :

-the set of system control functions,

-the set of subsystems control functions,

-the set of blocks control functions.

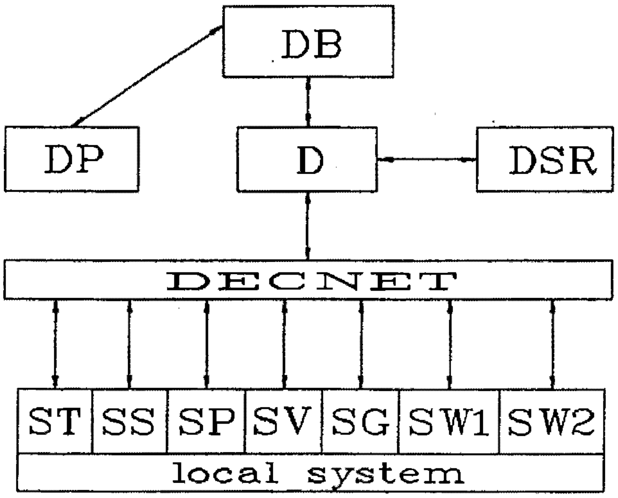


Figure 4. Software composition.

-the set of blocks control functions.

This elements are basis for local systems software. Graphic package is used for subsystems description special program. Changes of subsystems structures are made by means of screen picture editing. Special program transmits these changes to local system description file. Documentation about a system structure is printed at the same time.

Processing programs DP are intended for correction of errors arising on each element of a measuring line (censor-cable-registrator) and for realizing of users different algorithms. Correction programs include of "false" points liquidation, low and high frequency compensation. The example of distortion high frequency compensation arising in cable (length is 80m) is shown in fig.5.

User programs library contains arithmetic operations with signals including addition, subtraction, multiplication, Fourier transform etc. The software interface provides interaction of processing programs and data base.

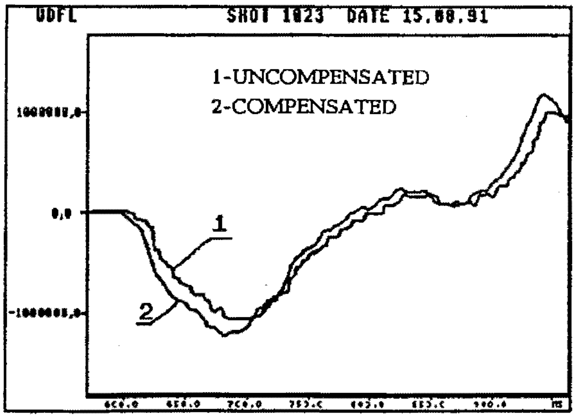


Figure 5. Example of high frequency compensation.

IV. CONCLUSION.

The tests of the hard- and software complex described in this paper have shown satisfactory correspondence to the principal installation performances. The insertion into hardware fast analog-to digital convertors is necessary for investigation of installation processes. It's desirable to realize the special corrective algorithms for the increasing of the measurement accuracy. The duration of data processing is a few minutes after each shot. The users can observe and process data at their working places by means of the ETHERNET. The structure of software let to adapt the complex for various experiments quite easy.

V. ACKNOWLEDGMENT.

The main uncommercial hardware have been designed in SNIIP. The authors are deeply indebted to Prof.S.S.Kurochkin, Dr. E.I.Rechin, Dr.Ig.I.Rogushin, Dr. Ju.A. Kruglov, Dr.G.N.Nicolaev for technical support and useful discussion.

VI. REFERENCES

[1] V.P.Smirnov, "Producing of high current electron beams", Sov. PTE N 2, pp.7-31,1977.
[2] G.Yonas, "Fusion power with Particle Beams",Sci.Am., v.239, N 5, pp.40-51,1978.
[3] E.V.Grabovsky et al., " "Angara-5-1" - superhigh electric power generator for investigation on inertial thermonuclear synthesis", "BEAMS-88"- In: Proc.7-th Int. Conf. on High Power Particle Beams-Karlsruhe, EC-5, p.333,1988.
[4] Z.A.Albicov, E.P.Velichov et al., "Pulse thermonuclear complex "Angara-5-1" ", Sov. Atomic Energy, vol.68, part.1, pp.6-35, 1990.
[5] S.S.Kurochkin, CAMAC-VECTOR Systems, Moscow: ENERGOIZDAT, 1981.
[6] V.I.Zaitsev et al., "Suppression of electromagnetic noise of pulse thermonuclear installation with power 9TW", Sov.PTE, N 4, pp.150-152, 1990.

A Control & Data Acquisition System for Photoelectron Spectroscopy Experiment Station at Hefei National Synchrotron Radiation Laboratory

Weimin Xu **Yimin Liu**
HESYRL USTC
Hefei, Anhui 230029

Abstract

The paper describes system configuration and software design. The system has the following features: flexible user interface, succinct control levels, strict protection and high intelligence. It can run EDC, CFS, CIS experiment modes very conveniently with SR light source. Its construction and design idea of the system can be applied to other data acquisition systems.

1. Introduction

Photoelectron Spectroscopy Experiment Station at HESYRL works from VUV to soft x-ray wavelength(10ev -- 1000ev) with resolution E/E 10^{-3} and flux 10^{10} phs/s[1]. The present photoelectron spectrometer is imported from VSW (Vacuum Scientific Instruments) Co., U.K. The software package also provided by VSW is mainly designed for regular light source and regular photoelectron spectroscopy analysis techniques, but not suitable for the special requirement of experiment modes with SR source. Up to now, having modified and developed the control system and software, the completely compatible system can not only carry out regular analysis techniques such as Auger Electron Spectroscopy (AES), UV-Photoelectron Spectroscopy (UPS), X-ray Photoelectron Spectroscopy (XPS), but also bring about Synchrotron Radiation Photoemission Spectroscopy (SRPES), Angle Resolved Photoemission Spectroscopy (ARPES), Near Edge X-ray Absorption Fine Structure (NEXAFS) and Photoelectron Diffraction (PED) with controllable SR source. We hope it will promote further research on surface science and material science.

2. Design principle

. Special requirement

The basic principle of photoelectron spectroscopy is to irradiate samples with monochromatic light source, causing photoemission from atoms or molecules, then analyze the electron energy and angular distribution and get the useful information. So a control system for photoelectron spectroscopy experiment is focused on two points: to change exciting source and acquiring methods of electron energy.

Exciting source

The system must control beam line availability, such as wavelength scanning for different users in different experiment techniques, zero order scan, photon intensity detecting and other analog signal measurement.

Four different spherical gratings and entrance/exit slits are installed in the beamline. It is demanded that the rotation of the grating and translation of stepping motor driven entrance and exit slits follow the Rowland circle.

Operating modes

Three operating modes with SR source must be inserted to the system: EDC mode, CFS mode and CIS mode.

.Design principle & implementation scheme

1) It is necessary not only to meet the special requirement above, but also to assume the most succinct man-machine interface. Beamline control is an example. Although there are so many parameters of intelligent motor controller, and the gratings and slits tracing movement is complex, there are just three necessary parameters in the user's interface to execute beamline scanning: grating number, initial and final energy of photon. The new added GPIB interface supports all beamline control.

2) The integrity of the original software manager level structure is to be preserved. All the extended periphery drivers are inserted into Software/Hardware interface layer. The Function layer is expanded and other layers are developed.

3) We assure a complete compatibility with original software both in general and details. In general, it includes the following fields: man-machine interface, experiment queue, file system, image display and protection measure, specific processing such as exciting source selection, experiment technique selection, parameters input and collection, the experiment condition recorded in data file, status display and so on are completely compatible.

3. General description

.General manager level structure

The system structure can be divided into the following layers: User interface layer, Analyzer, Scheduler, Function layer, Software/Hardware interface layer and Hardware layer, as shown in Fig. 1.

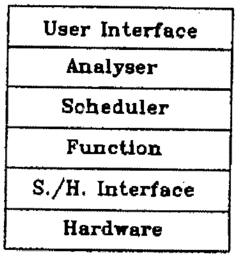


Fig.1 General manager level structure

User Interface layer

This level accepts and analyses preliminary user's input, it checks whether input is legal, displays peripheral equipment status and corresponding processing messages.

Analyzer

It makes a concrete analysis of user's input, then generates some legal results according to user's requirement and hardware environment. At last, the results are sent to the Scheduler.

Scheduler

It issues different function calls in the light of the results sent by Analyzer. If it must be supported by peripheral devices, it will transfer commands or parameters to peripheral devices and get status back through Software/Hardware interface. The calling skill is obtained by use of a special dictionary made of function pointers which just belongs to C language.

Function layer

It executes various concrete functions such as real time image display, reading and writing data files, queuing different experiments, running experiment and acquiring data. This layer and scheduler are the core of the system.

Software/Hardware interface layer

It is the lowest level of the software system. At this level, commands and data are transferred between system and peripheral devices. This layer includes all interface drivers and check of peripheral device status. If a certain device goes wrong or is not ready, a warning message will occur at the user interface.

.Hardware configuration

The whole system configuration is shown in Fig. 2. All the experiment equipment is connected to host machine through two main interfaces: Asynchronous communication port RS232 and GPIB port.

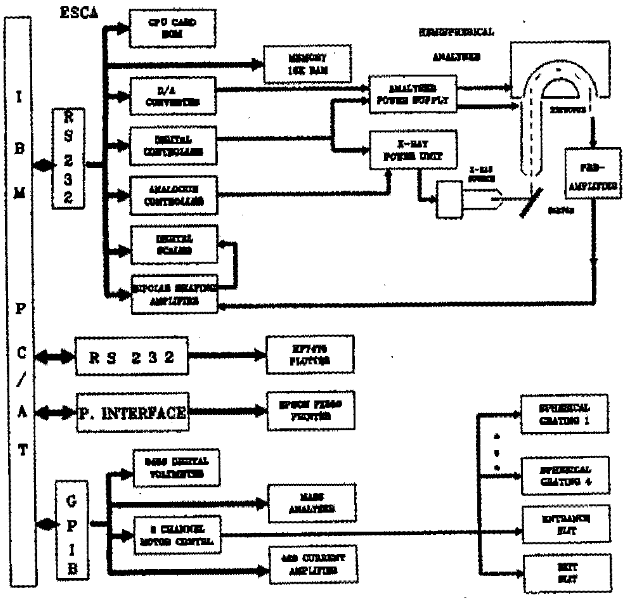


Fig. 2 Hardware configuration

RS232 serial interface

All devices of the photoelectron spectrometer and regular light source are connected to the special interface ESCA provided by VSW Co. and communicated with IBM-PC/AT via RS232. ESCA is an intelligent interface[2]. It contains CPU, ROM, RAM, converters, controllers, communication port and other I/O boards. The advantage of ESCA interface is to separate the software associated with hardware from data analysis at high level, so that users can cut out and extend software or hardware on the basis of different demands.

The photoelectron spectrometer has two kinds of electron energy analyzers: Angular Resolved HA50 which is equipped with HA5000 and HA300 controllers and Angular Integrated HA150 with HA5000 controller. The system provides Single Channel Detector (SCD) and Multi-Channel Detector (MCD) to acquire data. According to different conditions, MCD can be divided into three modes: Multi Slit Mode (MSL), pulse counting electronically variable slit mode (EVS) and Modulated Auger electronically variable slit mode (Auger)[3].

GPIB interface

All the devices concerned with beamline control, measurement and data acquisition are connected via GPIB interface. The essence of beamline scan is to control four different spherical gratings and entrance/exit slits by multi-channel stepping motor. The rotation of the grating and translation of the stepping motor driven entrance and exit slits follow the Rowland Circle to assure that the beamline has higher resolution over whole energy range. Besides, the mass analyzer also works via GPIB interface.

There are four CPUs in the whole system, 80286 in host computer, Motorola 6809 in ESCA interface, 8088 in stepping motor controller and CPU of multi voltmeter. These CPUs work in cooperation with each other and in charge of main control or sub control respectively.

.Software structure

Software system can be divided into the following function blocks as shown in Fig.3: Input module, File module, Image display module, Kernel processing module and Interface control module. The functions of each module are described in the following:

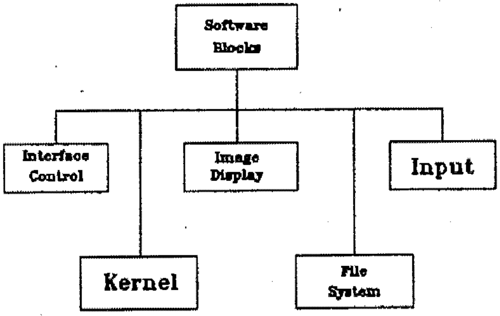


Fig.3 Software function blocks

Input module

It accepts user's input and belongs to User interface layer of manager level structure.

File module

It manages data files, generates or retrieves image files, builds or executes repeatedly command files. It belongs to Function layer.

Image module

It includes various technical processing such as to build man-machine interface windows, to display the acquired data, to form experiment spectrum, etc. This layer belongs to User interface and Function layers.

Kernel module

It executes user's command and implements concrete experiment task. It is permeated into all the layers in Fig.1 except for Software/Hardware interface layer.

Interface control module

It belongs to Software/Hardware interface layer. Besides transmitting common commands and data, it controls the two interfaces as follows:

ESCA Interface: Spectrometer control including to set up electrons' kinetic energy for energy analyzer, operation parameters, x-ray gun parameters, start energy, scan range and scan step, channels, dwell time of the detector for each channel and so on.

GPIB Interface: Beamline control including to set up parameters for each stepping motor, photon initial scan energy, scan range and scan step size, zero-scan, status display, photon intensity monitor and other analog signal measurement.

The layout of interface control is shown in Fig.4. Both INTERPRET and TERMINAL drop menu are man-machine interfaces associated with interface control. INTERPRET function dictionary belongs to kernel module.

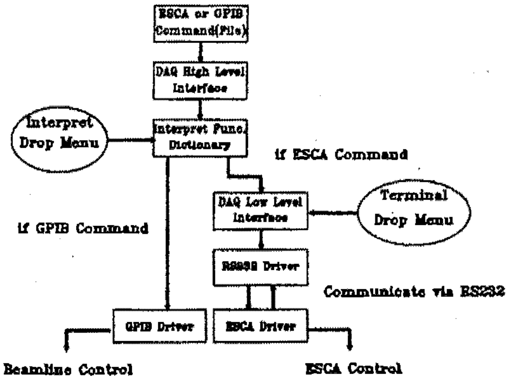


Fig.4 Structure of two interface control levels

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S05SRN11

.Experiment course

The system supports the following modes & techniques:

<u>Operating Mode</u>	
Electron Density Curve with fixed photon energy	EDC
Constant Final State with scanning photon energy	CFS
Constant Initial State with scanning photon energy	CIS

<u>Experiment Technique</u>	
Synchrotron Radiation Photoemission Spectroscopy	SRPES
Angle Resolved Photoemission Spectroscopy	ARPES
Near Edge X-ray Absorption Fine Structure	NEXAFS
Photoelectron Diffraction	PED
Auger Electron Spectroscopy	AES
X-Ray Photoelectron Spectroscopy	XPS
UV-Photoelectron Spectroscopy	UPS

Experiment Queue

To normalize experiment course, the system provides standard experiment queue table (Table 1) and experiment scan regions table (Table 2).

Table 1 shows Experiment Queue and its legal options.

Active	File	Regions	Technique	Analyser Controller	Detector	Source
y/n	•	#	EDC	HAC5000 or HAC300	MCD(MSL,EVS) or Scaler	SR
y/n	•	#	CFS	HAC5000 or HAC300	MCD(EVS) or Scaler	SR
y/n	•	#	CIS	HAC5000 or HAC300	MCD(EVS) or Scaler	SR
y/n	•	#	XPS	HAC5000	MCD(MSL,EVS) or Scaler	XRAY
y/n	•	#	UPS	HAC6000 or HAC300	MCD(MSL,EVS) or Scaler	UV
y/n	•	#	AES	HAC5000	MCD(MSL,EVS, Auger) or Scaler	E.G.

Table 1. Experiment Queue

Table 2 shows Scan Regions (take CFS mode as an example).

Active	Lable	Sweeps

CFS Mode					
Start	Span	Stop	Points	KEFixed	Dwell

HAC 6000						
Mode	Pass	+/-	Lens	Mag	Rev	Contact

MCD Detector			
Mod	Slits	Mode	Threshold

SR Source	
GratingNo	PhotonEnergy

Table 2. Scan Regions

Scan Regions table is composed of five parameter blocks. It will display a different parameter block according to the selected operating mode or experiment technique and experiment condition (in Table 1).

Experiment procedure

Users can set up parameters in the two tables above for different experiments and input command to start the system, then the whole experiment procedure will be run automatically. The acquired data are real time displayed on the screen, then are saved into memory. The data in memory are written to data file automatically after finishing scan for further processing later. If users have set up several experiments, it will run to next experiment as soon as finishing the last one. In addition, the system provides multi regions and multi scan functions to satisfy different user's needs. Users can deal with data in a special region repeatedly, so that the ratio of signal to noise and experiment result precision are improved.

It is convenient to build some non-standard experiment techniques at INTERPRET and TERMINAL man-machine interfaces.

.Data processing

Users can process experiment data off line by a separate data process software. The methods provided are the following: Data analysis (Smoothing, Background subtraction, Peak area measurement, Addition/Subtraction & Data re-scaling), Curve fitting and spectroscopy resolution, Deconvolution (Deconvolution by FFT or Iteration, Spin orbit partner removal by Deconvolution, X-ray satellite removal by deconvolution , Curve fitting by deconvolution)

All source files about 200 in number, are written in Microsoft C 5.1 and C 6.0.

4.Software features

The system adopts the following software techniques:

.Windows

All man-machine interfaces adopt advanced window techniques. The whole user layer can be located by keyboard and mouse. Users can input parameters, display status or execute action by main menu, drop menu, submenu, parameters input menu, command line and command file as shown in Fig.5. Generating, overlaying, saving and retrieving windows are implemented by reading/writing video memory directly.

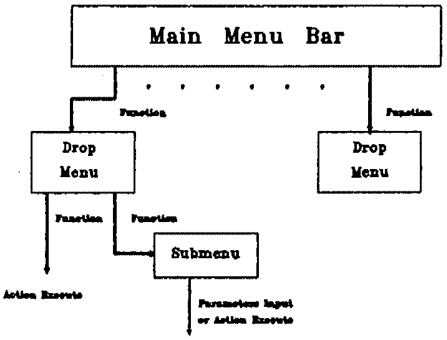


Fig.5 Menu levels

.File processing

File system has the functions of "save" and "retrieve". There are three kinds of files in the system.

The first one records experiment technique, experiment condition and experiment data. It can be further processed by a separate professional data processing software.

The second kind is an image file which displays real time during an experiment course. It ignores experiment conditions but preserves image attributes (screen color, image style, point coordinate...). It can be turned into image at any time but couldn't be processed by other data processing software.

The last one is a command file. The system can record command lines input by users and form a command file, then reserve it in order to run the experiment repeatedly at any time.

All the three kinds of files have one thing in common: the course of building, retrieving or executing files is implemented automatically. Users can input file name when message occurs.

.Real time display and image processing

The acquired data are displayed on the screen in separate (point) or continuous (line) style. The results are that a data file is formed and saved to disk automatically. The picture on the screen is a spectrum produced in the experiment course. In order to deal with wave peak or for other purposes, users can display the whole image, enlarge any part of the spectrum to full screen or move any part of the image to any location of screen. Of course, users can abort experiment and display course at any time.

.Protection measures

To avoid abnormal phenomena caused by mis-setting parameters and commands or unreasonable steps during an experiment course, some protection measures are adopted. The protection measures are as follows:

Protecting against mis-setting parameters

If users set incorrect parameters because of being new to the instrument, the system will match hardware set and correct it, so that the hardware can be well protected.

Long jump from error sites

Error sites, such as device error or device not ready, mismatch of experiment technique and experiment condition will make machine dead and abort experiments. In that case, apart from displaying various warning messages in time, the system can save sites and stack environment at any position where errors may occur. When an error is detected, the system restores stack environment and jumps back to the original site.

.Data structure

Among source files, all variables and data structures such as pointer, dimension, structure, union, enumeration, table, chain, tree are used alternately and flexibly. Thus, source files not only have good readability, but also support the modular design for the whole software system. For example, INTERPRET drop menu, one of the man-machine interfaces, accepts FORTH style command and parameters and matches professional dictionary to control devices connected to ESCA and GPIB interfaces. The data structure relative to this part is a table. Each table item is a structure, whose structure members are composed of command names and corresponding function pointers.

.Modularization design

In spite of the software system composed of about 200 source files, the structure is succinct and well readable. The method is to concentrate the functions which implement a certain kind of function, construct one or a few source files and form independent function modules. The external interfaces of such modules are minimized and the implementation details for other modules are transparent. For example, the window module includes all the necessary functions. When you would like to make a window, you just call the interface function of window module and pass attributes such as length, width, coordinate, color, title label, without considering implementation details such as video mode, window overlay, the location of video memory etc. The same principle is adopted in real time display, file system, exciting source control, analog signal measure and SR experiment modules.

.Portability

Although the control & data acquisition system has professional experiment techniques and was designed specially as a synchrotron radiation photoelectron spectroscopy experiment station, it can be referred to in other control & data acquisition systems. Especially, the structure parts of the software such as experiment course, image real-time display, data file generation, interface control levels and strict protection are important elements in a large data acquisition system. The system can be applied to other conditions if modified somehow according to hardware configuration and experiment demands.

Modularization structure and flexible data structure in software improve portability of programs. Users can cut out or extend the original software in the light of different demands. Most function modules can be separated to insert other software systems. Thus, programming work is reduced and efficiency increased.

Portability is shown in the Software/Hardware interface layer too. This is because the analysis schedule functions are at high level in software and don't involve low level at which functions relative to hardware are located. So, if you want to build a new control system with different hardware, just modify the function layer, insert Software/Hardware function layer, which matches new devices into the system. A new control and data acquisition system with different functions but of the same style will be created. We have designed and developed a new system for a photochemistry station at HESYRL with the same structure and have made a great success in this field.

Acknowledgment

The authors gratefully acknowledge Mr. W. J. Ellword of VSW Co. U.K. who wrote the original commercial software for his helpful advice on general design and software technique. The thanks will be also extended to Ms. Jin Liu of IHEP for her part in work on motor control, Mr. P. Xu and J. Li of HESYRL for their information about concrete experiment demand.

.References

[1]. Yunwu Zhang
Construction of Beamlines and Experimental stations at HESYRL
On Proceedings of the International Conference on Synchrotron Radiation Applications.
May 1989 P305
[2]. VSW Scientific Instruments LTD
ESCA Interface Unit Hardware, p4-7
[3]. VSW Scientific Instruments LTD
ESCA Interface Software Manual, 1989

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S06SA01

Overview of the Next Generation of Fermilab[1] Collider Software

Brian HENDRICKS, Robert JOSHEL
Fermilab, P.O. Box 500, Batavia, IL 60510, USA

Abstract

Fermilab[1] is entering an era of operating a more complex collider facility. In addition, new operator workstations are available that have increased capabilities. The task of providing updated software in this new environment precipitated a project called Colliding Beam Software (CBS). It was soon evident that a new approach was needed for developing console software. Hence CBS, although a common acronym, is too narrow a description. A new generation of the application program subroutine library has been created to enhance the existing programming environment with a set of value added tools. Several key Collider applications were written that exploit CBS tools. This paper will discuss the new tools and the underlying change in methodology in application program development for accelerator control at Fermilab.

I. MOTIVATION

Digital VAXstations running X-windows under VMS have replaced the PDP-11s formerly used to control the accelerators. This more powerful platform coupled with the demands of more complicated Collider operation led us to move from an approach of single application-per-need to using fewer applications that draw on a large toolbox of resources. Application programs are now viewed as hooks into a pool of integrated tools. At the same time we must maintain compatibility, while encouraging migration to new tools, for the existing application programs which number in the hundreds. This is done by providing calling sequences which are not too divergent from the existing ones. The new console platform also opened the door for the use of C as an application programming language. Calling sequences are often provided both in call-by-reference for the FORTRAN users and call-by-value for the C users.

II. OVERVIEW

All the new tools are layered on top of the older lower-level routines. These subroutines reside in a shareable image. This allows easy growth of a large number of routines without affecting the application programmer. This was necessary so that application development could be done in parallel with the maturation of the CBS environment. The tools handle file access, data acquisition, graphics screen management, window management, inter-program communication and error logging facilities. These utilities also provide their own logging and statistics that are viewable by the user during program execution. Tools that access centralized facilities, such as reading a database, cache information to reduce the load on the centralized processes and the network. Any of the tools with a visual interface follow standards. This provides a consistent user presentation to the operators. This is a more effective way to enforce user-standards, rather than administrative dependent approaches that have failed in the past.

III. DATA ACQUISITION

The first major component in the CBS utilities involves input and output to accelerator hardware as well as reading database information concerning that hardware. This involves reading, setting, controlling, and scaling values as well as handling alarms and miscellaneous device attributes. The previously available interface routines required separate requests for real-time raw data as well as stored data from the database in order to read or set data in engineering units. This required seven low level function calls to retrieve or set a single value in engineering units. In addition to the function calls, additional code was required to perform such necessary tasks as retrying data retrieval until the data is actually received. All of this functionality has been replaced by a single, simple function call. For lists of devices the procedure is only slightly more complex in that there is a function to build the list of devices and a second function to read or set the list.

The database information for scaling and necessary interface to front end software is cached locally. This reduces redundant database access and network traffic. The data acquistion routines perform the access and caching such that it is

---

[1] Operated by Universities Research Association for the Department of Energy

transparent to the application programmer. The cached information exists for the life of the program run. The database is updated so infrequently that stale cached data is not a problem.

The programmer and end user can peek at the data acquisition activity through an application that polls embedded statistics modules in the data acquisition routines. This peeker program runs on the console concurrently with the program to be analyzed. The peeker will display counts of function calls and errors. It will also show the number of devices and lists of devices being read and set as well as the frequency of retrievals. Additionally, timing statistics are shown for the data accesses being performed.

IV. FILE MANAGEMENT

There is a large group of applications that use shared read/write access files. A set of routines was created to help the programmer manage opening, reading, writing, and closing files in a simple, convenient fashion. As with the data acquisition routines, file operations were simplified by reducing the number of function calls required to accomplish them. A file peeker program was written that is similar to the data acquisition peeker. It displays numbers of function calls and errors. It also displays which file and record was last read and last written and any error associated with the access. In addition, timing information is displayed for communications with the central file server process.

V. SCREEN MANAGEMENT

The VAX console environment provides three X windows for each application program. These windows are managed by two manager processes that perform all the direct X protocol. This was done for compatibility with existing applications. The main window permits only character cell access. The other two windows allow pixel addressing and are used primarily for plotting. Screen management facilities were created for each of these types of windows.

The previous interface routines for the alphanumeric window provided little in the way of managing subwindows. There were window create and delete routines which simply saved and restored blocks of text on the screen. Anything beyond that was handled directly by the application program. This led to crowded and confusing displays and a myriad of user interfaces as programmers grappled with the problem of displaying a great deal of information in a limited space.

The new routines support input and output to multiple tiled or overlapping windows. Input and output is clipped to the window being addressed. In the case of overlapping windows, text written to an occluded character cell is saved and is refreshed when and if that character cell becomes exposed. These windows may be moved or resized by the user and/or under the control of the application program. Routines were also created to make it simple for existing nonwindowed applications to make use of the new windowed routines. In the future this transition may be made seamless by modifying the underlying I/O routines.

Support of windows with vertical scrolling capability was provided to release the application programmer from the limits of the size of the alphanumeric window. Routines exist to create and manage a scroll bar and draggable indicator requiring no support code from the application program. Lines scrolled out of a window are buffered, freeing the application from needing to remember scrolled text.

In addition to the basic window support, higher level screen management tools have been added. Menu and menu bar routines, numeric and textual input dialog boxes, logical dialogs, and message windows have been provided. There are also utilities to create and manage logical switches as well as to handle highlighting of text regions.

Since the pixel addressable windows are used primarily for plotting, a suite of plotting interface routines were created. The routines allow the programmer to define a plotting window in terms of fractions of the background window. After the region is defined, the scaling function and scaling limits can be selected for both axes. Plot labelling and plotting attributes can also be defined. An application program was also created to allow the user to enter window definition parameters and view the resultant window interactively. Once the desired window is constructed, the program can be instructed to generate the source code to create the plot window displayed. Additional routines have also been created to save plotted data and then to perform statistical and fitting operations on the saved data.

VI. PROGRAM TAPE RECORDING

One of the primary applications which had to be created for the operation of the Collider was the Colliding Beams Sequencer. This program carries out all of the steps to load the Collider with particles and bring them into collision. Some of these steps are simple and are contained within the sequencer, but some are more complicated and require the sequencer to invoke other application programs to perform the task. Modification of applications to run under sequencer control in the past has often been complicated and time consuming. It was felt that a more flexible means was needed for running programs under the control of another program. It was also important to have a method which would allow for easy modification. The facility implemented was a software tape recording system. A user enters the desired program in a special recording mode and then proceeds to perform the desired task manually. All the steps are recorded automatically in a file and are assigned a unique file name. The file can then be triggered by the sequencer (or the application itself), and the application program will perform the same steps originally carried out by the operator.

VII. ERROR HANDLING/LOGGING

Checking and reporting error conditions is one of the most important needs for application programs used for accelerator control. Operators need to know if settings and readings of devices are successful. In addition, it is useful to be able to reconstruct events that have occurred. The error handling routines check error codes, expand them to give text descriptions, and save errors encountered during execution of the program. Logging to circular log files can be done automatically by the error display functions. A program user can examine the log history in a scrollable window with many different text filter and search modes. The programmer can choose to incorporate a separate log for each instance of the program or a single log so that all uses of a program are shown in one log. Separate programs working together can also share a single log with time-ordered entries maintained by the logging utility.

VIII. INTERPROGRAM COMMUNICATION

To support the CBS utilities peeking activities and the Colliding Beam Sequencer communication, a set of interprogram communication routines were created. These use VMS mailboxes to allow queues of AST deliverable messages between indpendent processes. One simply creates a mailbox with an optional AST address and then sends or receives messages from that box. Programmers can now create program suites that can work independently or communicate directly with each other. The program used to smooth the Tevatron orbit will, if needing a lattice, start a program to generate the lattice. In turn it can then poll for the results. This allows large amounts of data to flow between cooperating programs at a high rate.

IX. UTILITIES

One of the CBS tools that demonstrates the integration of all the utilities is the Utilities Window. Through a menu driven interface the user can make screen copies, start plotting packages, invoke log displays and set timeouts for data acquisition and file accesses. The user can also interact with the error reporting buffer by clearing or viewing past messages. A window that allows display, reading and setting of accelerator parameters is also available. This parameter window is customized by the user for use with a particular program.

X. CONCLUSIONS

The results of recent accelerator studies indicate that the CBS approach has been successful thus far. The sequencer, orbit smoothing program, and the Tevatron ramp calculation and loading programs have been used successfully and have been well received by the user community. The consistency in user interfaces has allowed users to learn to use these complicated programs in a short period of time. In addition, programs have been developed using the CBS tools to create other applications not directly related to collider operation. Inexperienced programmers have been able to construct involved applications not only successfully, but also in short periods of time. The Collider applications have also been able to be extensively modified in response to user needs in a short period of time with little debugging.

The work continues on this project. There are many more topics to be addressed such as better handling of the data associated with the accelerator clock system and complicated table devices as well as context sensitive help. Ultimately, the goal is to bring all of this under the umbrella of a resource editor or code generation system to allow for even more rapid and error-free creation of accelerator applications.

XI. REFERENCES

[1] Cahill, K.J. and Smedinghoff, J.G. (1989). Converting the Fermilab Accelerator Control Consoles to X-Window Workstations, Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems, pp. 442-445

[2] Cahill, K.J. and Smedinghoff, J.G. (1991). Exploiting the X-Window Environment to Expand the Number, Reach, and Usefulness of Fermilab Accelerator Control Consoles, these proceedings

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S06SA02

IDEAS ON A GENERIC CONTROL SYSTEMS BASED ON THE EXPERIENCE ON THE 4 LEP EXPERIMENTS CONTROL SYSTEM

R. BARILLERE, J. M. LE GOFF, H. MILCENT, R. STAMPFLI
CERN, European Organization for Nuclear Research, 1211 Geneva 23, Switzerland

Most of the large slow control systems in the LEP collider experiments are distributed heterogeneous and multi-standard. But in spite of the appearances, they have a lot in common. From our direct experience on the L-3 slow control system and from the informations we obtained on the 3 other LEP experiments control systems we have come to the conclusion that it should be possible to build a Generic Control Package from which any control system could be derived. This software package is entirely based on relational databases and is intended to provide all the necessary tools to build a modular, coherent, easy to update and to maintain control system. Among other things this package should include user friendly interfaces, expert systems, and powerful graphic monitoring and control tools. This paper will present our general ideas about the realization of such a package.

1. Introduction

The need for large and dedicated Slow Control Systems for the High Energy Physics Experiments at CERN only became clear with the construction of the LEP collider. The large number of parameters involved in the controls as well as their wide distribution over the sites forced physicists to reconsider the entire organization of such systems. In this paper, the 4 LEP experiments control systems will be presented in some details followed by a summary of their main characteristics. Out of this study some general ideas which could be used for the elaboration of generic tools for future Slow Control systems will be presented.

2. The 4 LEP experiments Slow Controls

The 4 experiments are composed of many sub-detectors and several thousand operational parameters which have to be set and monitored in order to insure that the experiments are in a correct state for data-taking. These include quantities such as high and low voltages, temperatures, pressures and gas mixtures. The role of these Slow Control Systems is to monitor the experimental conditions locally and generate alarms in the main control room when faults are detected. Operators alerted by the alarm signals correct for the fault either manually or remotely via a computer. The systems are also used for the routine setup of operating conditions for example high voltages, general monitoring and logging of operating conditions.
Since the quantities to be monitored vary only on a time scale of seconds the slow control systems need not to have very fast responses. However, they must be very reliable and able to handle a large number of very different parameters.

2.1 The ALEPH Slow Control [1]

The ALEPH system is implemented in a number of distinct parts which are listed below:

1) Independent control and monitoring programs for each sub-detector run on VAX™ computers configured in a VAX™ cluster. These main displays of the currently monitored equipments allow operators to control the detectors and log the information to storage media.

2) All the sub-detectors slow control programs communicate with their equipment via a single server program running on a dedicated microVAX™ 3100 workstation in the cluster (fig. 1). The slow control server is the only program which communicates directly with the microprocessor system outlined below, which monitors and control the equipment. An interactive graphical status display showing the layout and status of the system is also implemented on this station.

3) Low-level monitoring and control functions are performed by programs running in a distributed system of microprocessors attached to the sub-detector hardware. They are capable of understanding, executing and replying to simple commands sent to them from the VAX server program.

4) Networks are used for communication between the slow control server and the microprocessors. The processors are connected to a number of UTI-NET

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S06SA02

local area networks and the VAX cluster to Ethernet. Ethernet-UTI-NET gateways have been installed to allow communications between the two different networks.

5) A large relational database describing the correspondence between the physical detector and the slow control system has been implemented on the VAX™ cluster. It is used extensively by the server to translate requests coming from the application programs given in terms of the detector high level commands into commands which will be understood by the microprocessor programs (low-level commands). The connectivity and the essential elements of the slow control systems are illustrated in figure 1.

The gateway software is written in C and runs on a Motorola 68010™ processor under the OS-9 operating system. Its role is to synchronize the transfer of messages between the two data networks which have very different physical characteristics. The program is loaded on resetting the system from files residing on the VAX™ cluster Network disk.

The central element of the ALEPH slow control system, as indicated before is a relational database which describes all the aspects of the system. It is one sub-section of the online database which also contains the descriptions of the Fastbus system and the readout configuration. The entity-relationship model is used in the design of all the databases [1]. In such databases, elements in a system are represented by tables. A particular instance of an element occupies a row in a table and each attribute which describes the elements occupies a column. Furthermore, the description of each element may be supplemented by specifying its relationship with other tables in the database or in other databases. The advantage of this type of structure is to avoid the need of repeating information unnecessarily and thus to reduce the possibility of conflicting entries and enhancing the data integrity. Entity-relationship diagrams have been used during the design phase of the database and provide a convenient means of displaying the database structure.

A graphic status color display showing the current status of the system is running on a colour workstation in the VAX™ cluster. In addition it allows inconsistencies in the database to be spotted more easily. The display has been made interactive and layered to cope with the large amount of informations involved.

The display program is driven by a graphical database derived from the slow control one using the ALEPH graphics package [1].

2.2 The DELPHI Slow Control [2]

The DELPHI slow control system is composed of three major logical components. The G-64 microprocessor systems, very similar to the one of ALEPH, provide the interface to the hardware to be controlled or monitored. The intermediate level software running on microVAXes and VAX™ stations, called the equipment computers, which handles all the low level processors belonging to a single detector. And finally the high level software which performs the overall control is running on the central control computers (VAX™8700 and VAX™4000).

1) The G-64 Systems

Each system is equipped with a Motorola 6809™ processor, RAM/ROM memory and commercial cards like ADCs, DACs and HV controllers. The communication with the intermediate layer is performed by a G-64-cheapernet interface board which has been developed at CERN. The main functions of the standardized software running in these systems are the following: at startup, the software sets the value of all physical channels to preselected ones which are stored in the VAX™ and downloaded to the G-64. During normal running operation, it monitors the values of all the channels in a tight program loop and reports the significant anomalies and the alarms to the program running in the VAX™ equipment computer. Also during normal operation it receives and carries out instructions coming from the equipment computer like, ramping up or down the high voltage of a given device.

2) VAX equipment computer software

The software running in each sub-detector equipment computer handles all the aspects of the communication between the G-64 layer and the master central layer. It receives actions from the top layer and distributes them to the interested G-64 system. It also collects all the alarm messages and anomalies from the bottom layers and forwards them to the top layer. Finally it updates the central DELPHI database with the values of the parameters and their status. At startup this program also performs the downloading of the preset conditions of each parameters from the database to the G-64 targets. Standard software has been developed by DELPHI to perform these functions. Its form was determined by the choice of the automated control system based

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S06SA02

on a CERN package called the State Management Interface (SMI) of the Model software [3].

3) Central Control and Interfacing Software

This layer also uses the SMI package but applied at a higher hierarchical level than before. This SMI single program interacts with all the different sub-detector control systems. No fortran processes are involved. A central operator's display allows the overall state of the experiment to be viewed from a VAX™ station 3100 using the Motif™ Graphics system. On the basis of the information displayed the operator can decide to start or to stop the run. The messages generated by the sub-detector control systems are also displayed on this graphic interface. A general view of the system is presented in figure 2.

2.3 The L-3 Slow Control [3]

The L-3 system is also modular and is composed of four different layers (fig. 3), each of them performing very distinct functions:

1) The BBL3 Layer

This a non-computer controlled hardwired layer dedicated to the high level alarms whose consequences would be of major importance for the experiment if the necessary actions were not taken on time. Each of these hardware box offers a maximum of 48 isolated inputs and 32 outputs specially designed to perform important actions as to power off a control room or a piece of equipment which is not operating well. A manually programmable matrix allows any input combination to perform any number of distributed actions. The all system is being independently read out by the computer of the detector which owns the box and by the second layer of the slow control system described after.

2) The local Slow Control Layer

This is a typical local control system which reads out the parameters of a given piece of hardware, performs a local analysis of its behavior, reports to the central layer for alarms or new data and takes local actions which are within its range. Depending on the equipment this software is running either on a VAX™ computer or on a dedicated PC or a VME OS-9 system. It has been designed to be as little equipment dependent as possible and is reasonably data driven.

3) The central Slow Control Layer

For safety reasons, this very important layer is running on a standalone microVAX™3200. It houses the database which describes all the parameters the L-3 slow control is responsible for. Its main functions are to collect from the network all the informations related to all the equipments like alarms or digital monitoring data blocks. After every update of the shared memory which holds the present status of the experiment, an expert system is triggered to analyse the situation. It takes actions when necessary. These actions will be broadcasted to the lower level computers where they will be physically executed. In case of time out due to network problems or to other reasons the expert system may take higher level actions if it has been foreseen in the database. Otherwise, the serious situation in which the system is, will be immediately reported to the operator. In case of hardware alarms where the actions are taken independently of the software, the system will report the new status of the equipment of concern and will not proceed any further. At each modification of the content of the shared memory, a trace of what happened is written on a daily file. It will also appear to the operator as a structured message which has been preprocessed from the information contained in the database. A graphic display system running on various Macintosh™ computers located in the main control room and in various places at CERN will immediately be informed of any changes.

4) The Color Graphic Display System [4]

This sophisticated graphic display has been entirely written in Object Oriented C making use of the THINK C™ library of the SYMANTEC company. It allows an operator to view the status of the entire experiment just by clicking into sensitive boxes whose colour represents the worse status of all the parameters or windows they give access to. The all window tree mechanism is housed in the database of the third layer as well as the description and location of all the parameters to be displayed. At startup the application inquires to the central slow control the date and time of the last modifications in the database. If they differ from the ones corresponding to its local copy, it will request an update over the network. It will then build the application objects from these new static data. It is important to mention that this graphic system is entirely data driven and does not need any maintenance. The application software is itself available on a Macintosh™ server if needed and does not have to be resident in the macintosh™.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S06SA02

2.4 The OPAL Slow Control [4]

Unlike the three other systems, the OPAL slow control system is not a layered hardware system. A series of VME OS-9 creates all connected via Ethernet constitutes the backbone of the system. Each of them is dedicated to various tasks regarding a part of the experiment. A Macintosh™ using the Hypercard™ software of the Apple Company is directly connected to the VMEBus and allows a local graphic display of the parameters under the control of a given crate. The software which performs these tasks is organized in three layers presented in figure 4 and described below.

1) Human Interface

There is a variety of possible interactions with the control system, such as sending a command and retrieving an answer message. A comprehensive and powerful human interface to handle the whole OPAL detector has been implemented on an Apple Macintosh™IIx, using the Hypercard™ tool. It uses graphics to interact with the operator. SC_MENU is an alphanumeric menu-driven interface which runs on a terminal. Run control is a dedicated data acquisition program, while SC_ALARM displays warnings and alarms for the whole system. Any computer on the network can be used as a sending or receiving node; however, a clear separation is made between "harmless" commands (e.g. to get a status) and actions on the detector. The later are normally not enable from remote sources.

2) Skeleton

At boot time, SC_PROC starts all the programs in layer 2 and supervises them. They are all written in C language. Commands are sent to the program SC_FILTER, which checks their validity, writes them in a log-file if required and passes them to the appropriate application in layer 3. This program executes the command and may them send an answer message back to SC_ROUTER which has already been notified where the answer should be sent to and which passes it on. Unsolicited output of the application programs, such as errors, is accepted by SC_ERROR, which logs it in a file and, if required, passes it on. Automatic corrective actions are handled in the same way and sent to SC_FILTER.

3) Applications

An important consequence of this clean separation into three layers is that the same application programs can run in both stand-alone mode (e.g., for debugging) and in an integrated mode with the entire system. They can be written in any language. The application programs are also supervised by SC_PROC, which starts and stops them on request. A typical example is the program which reads the ADC of common controls. This program is driven by a table which gives the characteristics of individual channels such as their description, warning and alarms limits, actions to take, etc. It compares all parameters with their nominal values every 5 seconds. Asynchronously, it also accepts commands to show the status, change default values, disable actions, etc.

3. Main characteristics of the 4 systems

Due to the large number of parameters to be controlled and their wide distribution, the 4 slow controls have adopted the layered and distributed system principle. For the same reasons dedicated and unspecialized slow control programs have been written to reduce redundancy in the code. In most cases special efforts have been made on the design to provide systems as close as possible to an ideal data driven slow control. But unfortunately, in most cases it has not been possible to achieve this goal all the way down to the lowest control layers. The importance of databases in such systems have been clearly understood and will be the key part of future slow control systems. These four systems have been running since the LEP collider started in July 1989 and are expected to operate until late in the decade. In the four cases major upgrades of the hardware of the detectors in the experiments are expected. These will force every slow control system to follow up closely. But due to the limitations of some of the software and hardware, and also due to the lack of modern software engineering tools when these systems have been created, it is to be anticipated that major maintenance efforts will have to be done.

4. A possible Generic Control System

To satisfy fully the data driven constraints, such a system should entirely be based on large relational databases where tools like CASE tools and entity relationship models would insure a full coherency between the different tables.

As illustrated in figure 5, a set of external software engineering tools like a Configuration Manager and a Coherency Manager could be implemented to help the user to perform the design and the upgrades of his slow control system by following strict and formalized rules. Furthermore, with no extra efforts an up to date formatted documentation would be readily available. The data blocks and their definition would then be extracted from the database and distributed over the network to the target computers where the necessary updates would be accomplished. At every level of the system, tools would be provided to allow an automatic implementation of the newly received data blocks. An acknowledgement mechanism in relation with the Configuration Manager would also have to be designed to reduce the system instability during these transition phases.

The internal organization of the database and the corresponding software layers are presented in figure 6. Five main layers can be identified.

1) The Local Acquisition and Command Layer

This layer is dedicated to perform the typical and basic operations of any local control systems. Namely, read the data from the hardware, interprete them according to the local Acquisition and Command database content and take the actions which could be requested by the Hardware Control layer.

2) The Hardware Control Interface.

This layer takes locally the decisions which are necessary to keep the lowest layer in operation according to the Hardware Control database, for trivial decisions, and to the Software Control layer for more complicated ones.

3) The Software Control Interface

All the actions which can be internally taken and which are independent of the other control systems will be handled at this level. Furthermore this level is responsible for the synchronization between the communication level and the three bottom layers during the updates of the system.

These three layers are completely autonomous and can perform a full local control of a piece of equipment in a stand alone mode. A optional expert system could very well be included in the third layer when needed by the designer.

4) The communication layer

All the data and alarm transfers to the central control system is taken care by this layer. It is also responsible for the temporary storage of the new data blocks issued by the database until the full update mechanism has been completed.

5) The Central Control Layer

This layer holds a copy of the values of all the parameters controlled by the experiment. It will forward the actions requested by the operator to all the sub-systems involved and later report for success or failure. No actions issued at this level will take effect at the level of a local system, if it has not previously been implemented in its Software Control Interface layer. Since this can only be done by the person in charge of the equipment, this avoid spurious and not commissioned actions to take place.

A distributed color graphic system will display either the global status of the experiment or the local status of a piece of equipment by interrogating the central control Layer. The same format and philosophy should be provided for both local and global displays. The upgrades of this system should be handled through the configuration manager.

5. Conclusions

In this paper, it is not possible to go into more details into the organization of this generic systems. But we believe that with the support of industrial software products like CASE tools and others it would be possible to provide the slow control designers with a set of tools which could help them greatly in the setup and maintenance of their systems all along the years of operation.

References

[1] The ALEPH Slow Control System, S. Wheeler et al., ALEPH 90-67 DATAACQ 90-08.

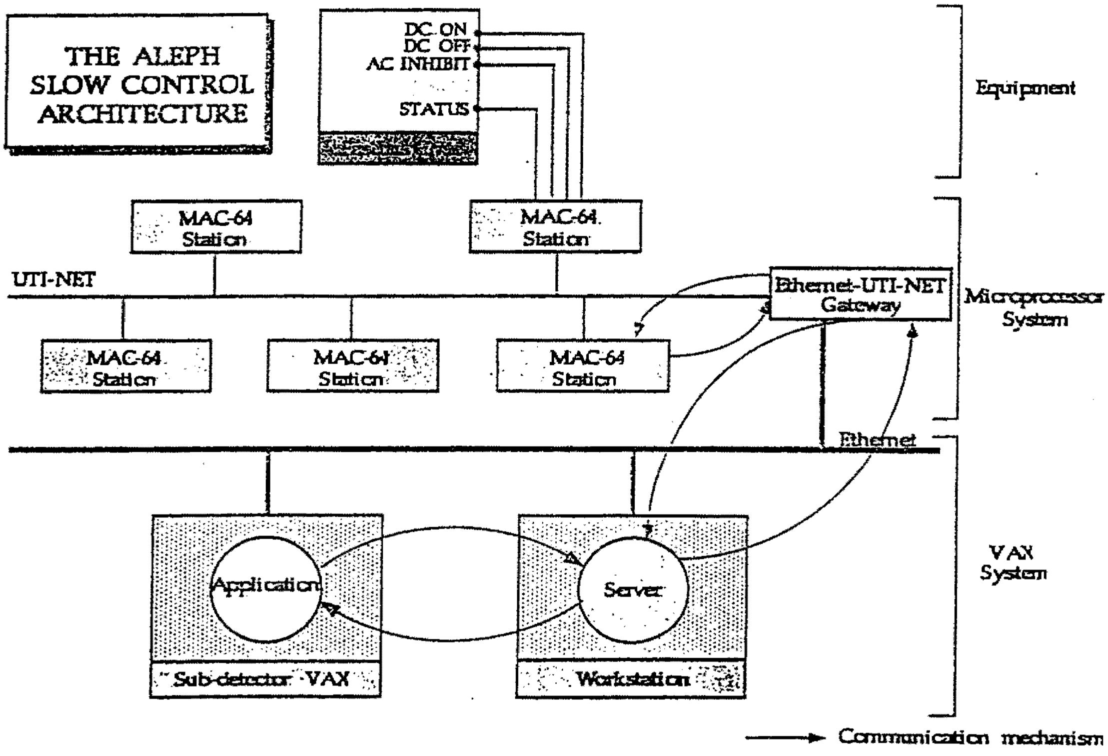
[2] The DELPHI Slow Control System, R. Sekulin at al., DELPHI Internal Note.

[3] The L-3 Slow Control System, J. M. Le Goff et al. Paper presented at the Conference on Software Engineering Artificial Intelligence applications in High Energy Physics, LYON March 1989.

[4] The ARGUS graphic system, J. M. Le Goff et al. Paper presented at the CHEP91 conference, TSUKUBA March 1991.

[5] The Control system of the OPAL detector at LEP A.K. Amundsen et al. NIM A293(1990)145-147.

Figure 1: The ALEPH SLOW CONTROL ARCHITECTURE



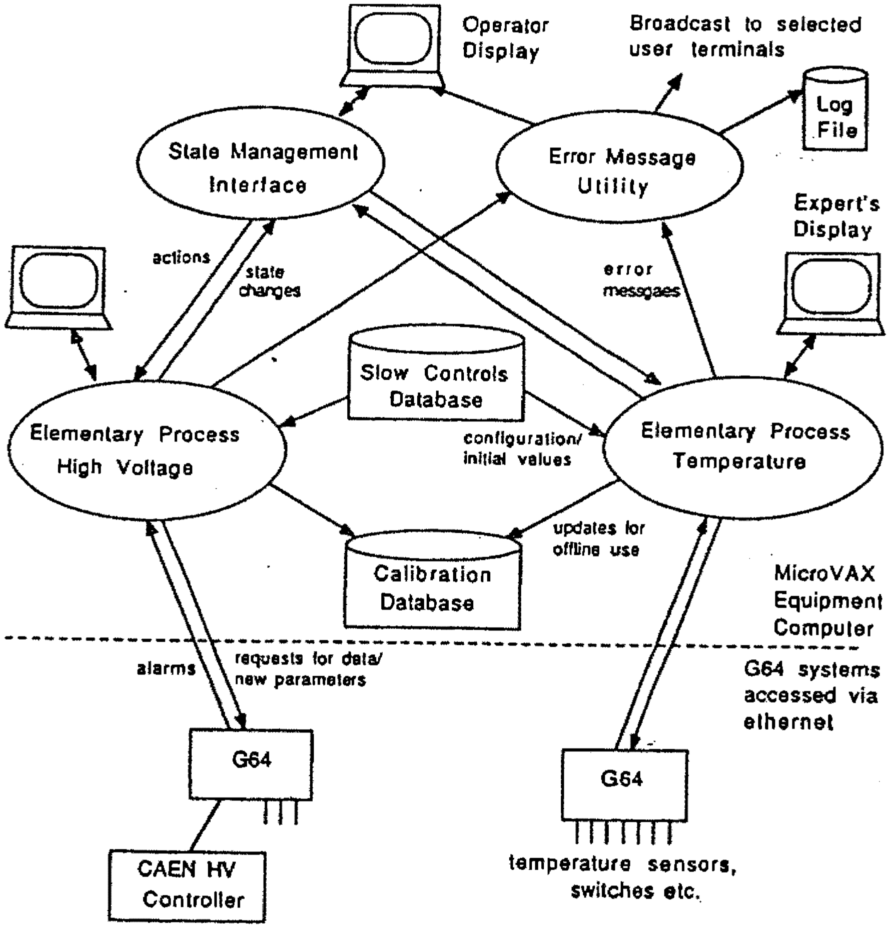


FIG.2: DELPHI SLOW CONTROLS SYSTEM

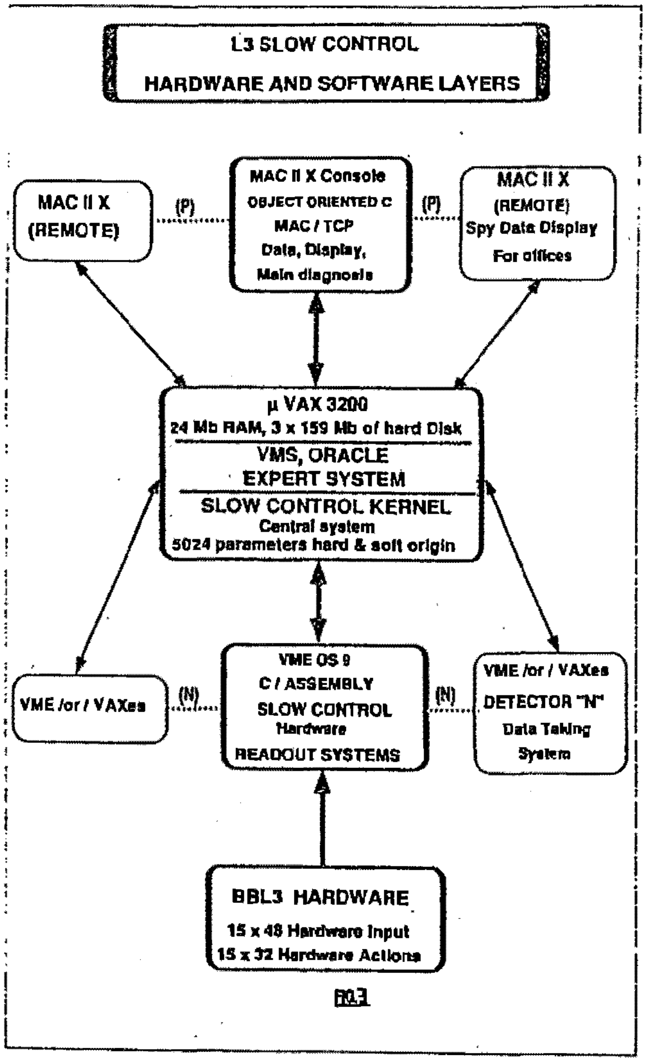


FIG.3

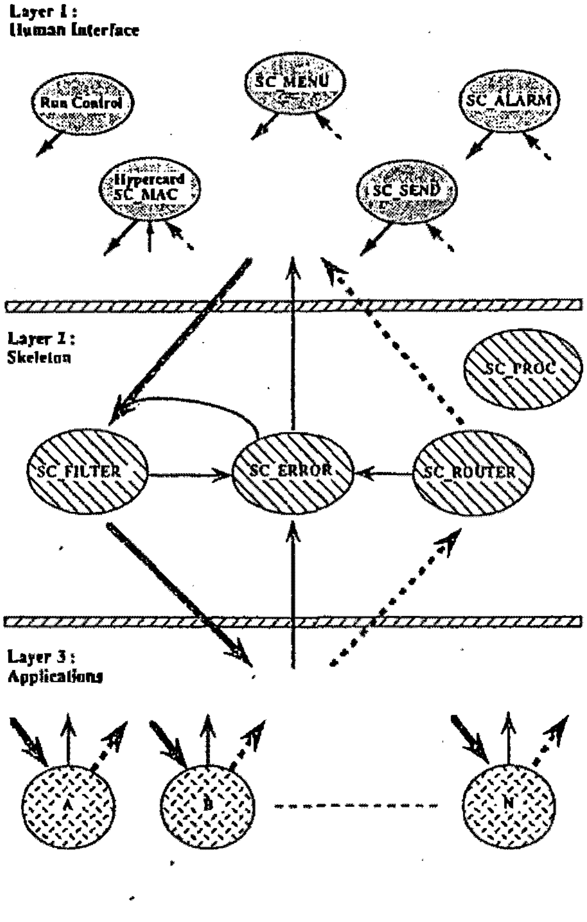


Figure 4: The OPAL SLOW CONTROL SOFTWARE LAYERS

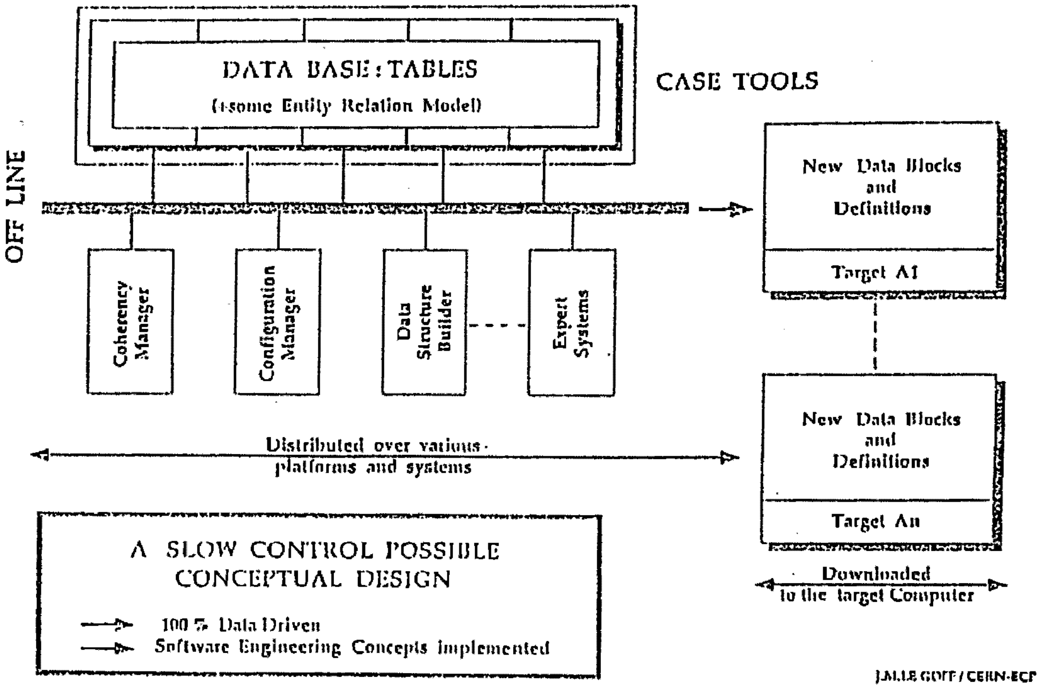


Figure 5: The GENERIC CONTROL OFFLINE Configuration

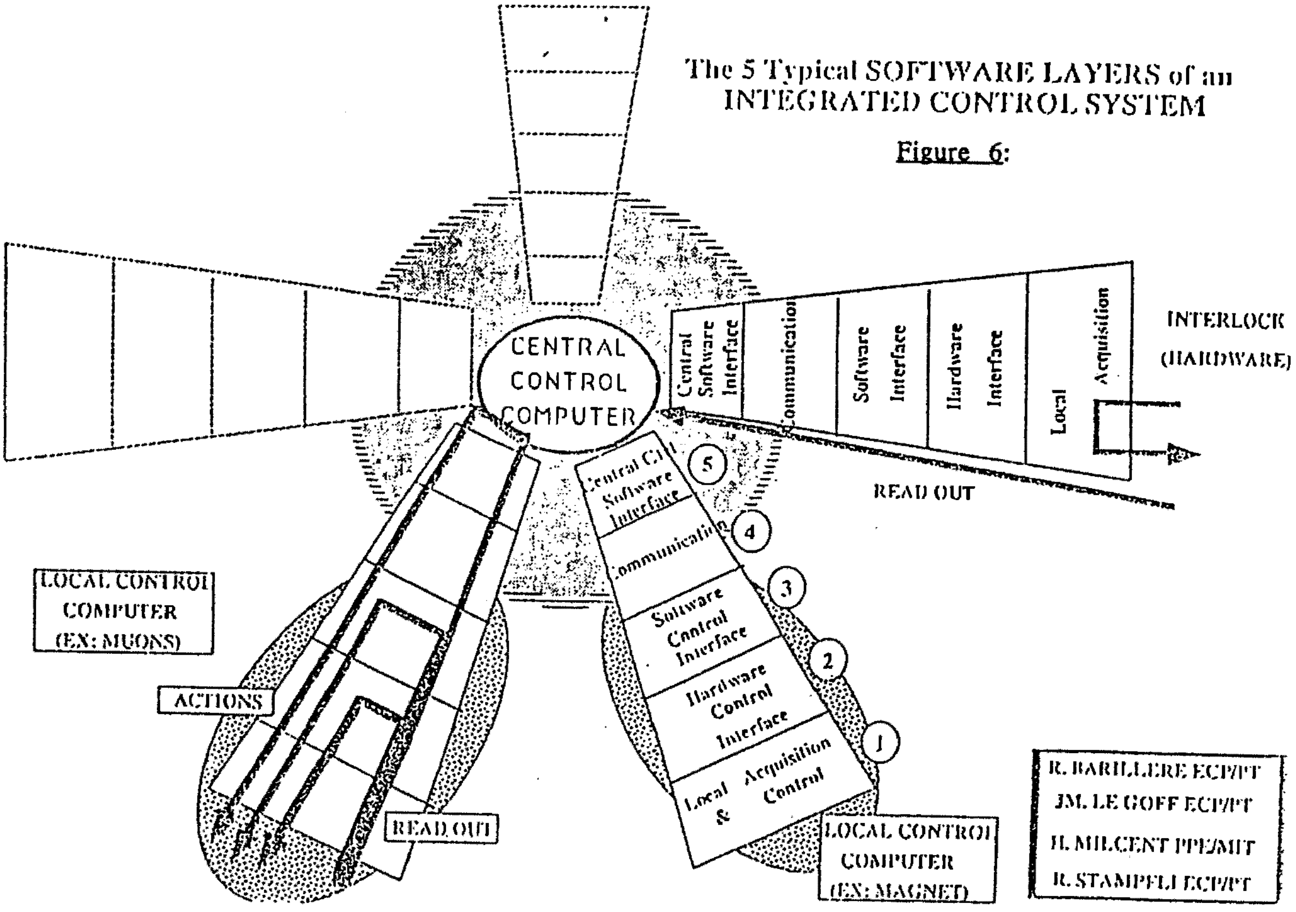

The 5 Typical SOFTWARE LAYERS of an INTEGRATED CONTROL SYSTEM

Figure 6:

The LEP Alarm System

M. W. Tyrrell
CERN - European Organisation for Nuclear Research
SL Division, Controls Group
CH-1211 GENEVA 23, Switzerland

Abstract

Unlike alarm systems for previous accelerators, the LEP alarm system caters not only for the operation of the accelerator but also for technical services and provides the direct channel for personnel safety. It was commissioned during 1989 and has seen a continued development up to the present day. The system, comprising over 50 computers including 5 different platforms and 4 different operating systems, is described. The hierarchical structure of the software is outlined from the interface to the equipment groups, through the front end computers to the central server, and finally to the operator consoles. Reasons are given for choosing a conventional, as opposed to a 'knowledge based' approach. Finally, references are made to a prototype real time expert system for surveying the power converters of LEP, which was conducted during 1990 as part of the alarm development program.

I. INTRODUCTION

The Large Electron Positron Collider (LEP) was constructed during the years 1983 to 1989 and is situated in a 27km tunnel of diameter 3.8 metres at a depth varying between 50 and 175 metres. It contains 4 experimental halls situated symmetrically around the ring of size approximately 80 metres long and 23 metres in diameter. From the very outset it was decided to survey the whole complex both for personnel safety and equipment status by 1 alarm system due to the sheer size of LEP and cost of system installation. This task was given to an 'Alarm Team'. A major milestone for the project was the use of a complete prototype for the 'LEP Injection Tests' in July 1988. By 1989, in time for 'LEP Switch On' the system had become stable and operational, but with an incomplete coverage of the complex and only a rudimentary display for information presentation. A continued development concentrated on improving the Man Machine Interface (MMI), and extending the scrutiny of the surveillance system while investigating alternative techniques to improve the overall system.

II. THE ALARM SYSTEM PROJECT

A. Definition

The alarm system can be thought of as a window through which operators can view the status of any part of the process. Here the process concerns the whole accelerator, equipment associated with personnel safety and the control system itself. By definition, if there is nothing wrong with the process, it is assumed that the overall state is good, and therefore no alarm information is presented. On the other hand, whenever a piece of equipment does not work, or an abnormal state is detected, a description of this situation should be passed to the system. This description is termed a Fault State (FS) and covers both warning and alarm situations. The alarm system concerns the acceptance, treatment and display of these FS's.

B. Organisation

The project began in earnest in 1987 and has been continually staffed by 1 permanent and, on average, 3 temporary personnel. During the 5 years of system design, implementation and development, 16 temporary personnel, each working on average 1 year have contributed 18 man years to the project. These people were all trained in computer science, apart from 1 who was an experimental physicist. The permanent member managed and coordinated the project which was divided into 4 areas: display and presentation of information to operators, a Central Alarm Server (CAS), an interface to the users responsible for the equipment, and a database.

As part of the organisation of the LEP project, it was stated that each equipment group should be responsible for its own equipment surveillance. It was envisaged that the main part of this task would be done in the equipment groups' local control environment called Equipment Control Assemblies (ECA's). Unfortunately, in reality little surveillance of equipment was implemented at that level. This required that the interface between the alarm system and the equipment groups, in many cases passed beyond the ECA right down to the level of the equipment. For this reason it was even more important to define exactly where each line of responsibility was drawn. In practice this was always done at the level of a database definition, either at the FS description level or a combination of a command/response definition to acquire an equipment state and its corresponding FS definition.

C. Influencing Factors

The main influencing factors were the following:

1. At the very beginning of the project various groups at CERN were evaluating the possible uses of Expert Systems (ES). At one time an 'Expert System Interest Group' was set up which encouraged a free exchange of ideas and helped keep abreast of developments. Although nothing of practical value resulted for LEP, one area, a project in the Controls Group of

the Proton Synchrotron Division at CERN [1], did develop, and continues today.

2. Throughout the design and implementation of the alarm system, close collaboration was maintained with the group responsible for the safety system in the experimental zones, namely the General Surveillance System (GSS) [2]. Both system designs were exposed to the conventional approach, but early on in the GSS project, an ES shell developed at the Electricite de France, called Genesia [3], was evaluated and finally used. The alarm team decided against its use for the following reasons: original lack of portability, lack of internal control of inferencing, limited interface to the outside world, constraints on the use of variables, rule set generation time very long and no temporal reasoning facilities. One conclusion of this work was the realisation that such an approach encouraged the use of 'rules' to define logical relationships but that the maintenance of a large interrelated rule set is by no means trivial.

3. The environment in which the alarm system had to run was also a strong influencing factor. At the equipment level, equipment groups built their own ECA's containing different processors and operating systems. This included the 4 GSS systems, which for connection purposes were considered as ECA's. UNIX was chosen as the operating system for the control system with 'C' the programming language. To-date various elements of the alarm system run on 5 different platforms covering 2 operating systems including 4 flavours of UNIX.

4. The LEP machine falls under the jurisdiction of the French authorities which consider accelerators as nuclear installations. They must conform to strict safety regulations with respect to safety of personnel and particularly to radiation exposure. This meant that the design of the alarm system had to provide the necessary features required by these regulations, including formal FS definitions for safety, redundant FS transmission paths and the use of a fail-safe power supply network.

III. CONTROL SYSTEM ARCHITECTURE

The control system [4] is based on 3 levels of processing. At the lowest level are the ECA's of the equipment groups. They are connected to the next level using the MIL-1553-B multidrop bus. The controls group is responsible for the interface between the ECA's and this level, including: the hardware and software interface; a command/response protocol for control; a local name server; and an alarm channel for passing alarm information. The intermediate level consists of Personnel Computers (PC's), 386 machines, known as Process Control Assemblies (PCA's). These act as concentrators of processing power both for equipment control and alarm handling. There may be up to 10 such machines at one LEP site. Each machine is either connected to a local Ethernet segment and then a Token Ring (TR) (IEEE 802.5) or directly to a TR which in turn connects to a Time Division Multiplexer (TDM) (CCITT-G700) system for long distance transmission. The TDM connects the various PCA's to local TR networks in the different control centres. It is on these local networks that the third level of processing exists in the form of consoles for control and the display of FS information.

IV. FAULT STATE CHARACTERISTICS

A FS has been defined as something wrong with the process. It is described as a triplet: Fault Family (FF); Fault Member (FM); and Fault Code (FC). The FF is a collection of similar parts of the process, exhibiting similar FS's. A FM of this FF is an instance of one of these parts of the process. The FC describes the problem. For a perfect FF, all FC's will apply to all FM's of that FF. As an example, in the services environment consider the fire detection system. All fire detectors are grouped under the FF 'FIRE_ZONE'. The FM defines uniquely each fire detector or circuit. The FC describes the problem: smoke-detected alarm level; smoke-detected warning level; detector under maintenance; detector out of service; and detector fault. For the machine, consider the power converters. They are all grouped under the FF 'POWER_CONVERTERS'. The FM is the name of each individual power converter and the FC describes the problem: faulty; timing error; local mains variation; tune loop control error; ECA system reset; spike detected; etc..

Although this triplet definition defines the FS uniquely, it is not sufficient for alarm management or operators receiving these FS's. For this reason a 14 field, character string was formally defined. The first 7 fields are obligatory and consist of the following: a string version number; the triplet; a flag indicating whether the FS has just become active, or that it has now terminated; description of the problem; and a priority indicating severity. The remaining optional fields enable a user to define more precisely the FS. A formal name was given to this string namely: User Ascii Version 1 (UAV1), which is used to describe FS's in ECA's , PCA's and any other computer performing surveillance.

An important aspect of a FS is time. A FS is considered to be 'active' during the time it has a status 'true'. Such a state has 2 times associated with it: the time at which it became active; and the time it terminated. During the interval between it is referred to as an Active Fault (AF) and is added to a list containing all AF's, this list being called the Active List (AL). An AL is maintained at each level of the alarm software. It represents the current state of the process as surveyed by that software level.

Not all temporal aspects of problems are covered by the concept of an AF, since there are very important 'events' which take place at an instance of time, and thereafter have no further meaning. Examples of these are: an ECA microprocessor 'resetting'; a spark in an electrostatic separator; and a software task 'timing out'. These situations are treated differently by the system. They are described as 'Instant Faults' and have a corresponding identification flag in the UAV1 string. Since they only have 1 time stamp associated with them, they cannot be part of an AL. Instead they are passed through the system and finally offered for display.

V. HIERARCHICAL STRUCTURE OF THE ALARM SYSTEM

LEP has 8 access points distributed equally around its circumference. These areas are used for personnel access, concentration of general services, and points of localisation for the control system; in addition the even points house the 4 underground experiments. Control centres for LEP are distributed widely over the CERN site, with the machine control room situated on the Prevessin Site, and the Technical and Safety Control Room (SCR) situated on the Meyrin Site, some 3km away.

This geographical layout lent itself to a hierarchical alarm system approach both in terms of hardware and software. At the lowest level, ECA's provide either FS or status information to the PCA's; the 4 GSS systems of the experimental zones mirrors this behaviour. Included at the PCA level are computers responsible for the surveillance of systems, not connected to ECA's. This layer then passes the FS's to the CAS, where they are centralised and grouped according to operators' areas of interest, such as machine operation, safety etc.. Finally, if a console has been initialised to receive one or more of these FS areas of interest, the CAS will send all corresponding states to that console, where alarm software will display them.

VI. ALARM SOFTWARE WITHIN THE PCA'S

A. General Structure

There are 3 layers of software within a PCA [5]. At the lowest level there is the Low Level Alarm Server (LLAS). The middle layer concerns Surveillance Programs (SP's) which can be divided into 2 sub-layers: that dealing with Standard Surveillance Programs (SSP's) which always exist, and that dealing with User Surveillance Programs (USP's). Finally at the top there is the Local Alarm Server (LAS). All these layers communicate in a standard way using messages and all processes are either active, or waiting for a message and or a timer: they never die.

B. Low Level Alarm Server

The LLAS receives FS's from 'intelligent' type ECA's via the alarm channel of the multidrop bus. They may be the complete description of a problem detected and transmitted by the most intelligent class of ECA's, or a pseudo FS from a less intelligent ECA simply indicating that a change in state has taken place. In the latter case, the pseudo FS may contain the current state as data, which must then be interrupted at the SP level. The role of the LLAS is first to transform the received FS into a standard UAV1 string, then to extend it into a Standard Ascii (SAV1) string through the addition of the computer name, SP name and arrival time. Finally, using the FF and an internal correspondence table, generated from the database, the LLAS directs the FS to its corresponding SSP at the SP level.

C. Standard Surveillance Programs

A SSP deals with all instances of one type of equipment connected to a PCA. This allows logical analysis of FS's concerning a particular equipment area, which is usually completely different to that required for another equipment area. This leads in some cases to dedicated software within certain SSP's which is well separated from the standard software.

To improve the maintainability, the ability to accept changes, and to provide a 'template' for a SSP, each SSP is data driven using 2 standard flat files generated from the database. These files are used to build internal tables at initialisation. One contains all possible FS's considered valid for that SSP, together with a 'LEP Mode Mask' for each FS which indicates the applicability of each FS under each machine mode. This table is used for: the reduction of FS's within or across FF's; the treatment of oscillating FS's; and the possibility to inhibit, on request from a control centre console, the transmission of particular FS's. In all cases it is possible for operators to view remotely the underlying FS details. The second table contains command/response information necessary to contact the attached ECA's, and in some cases the correspondence between bits in an equipment status word and their FS description.

SSP's attached to intelligent ECA's use these tables to request periodically the state of each relevant ECA to verify the consistency of their AL's, rectifying any discrepancies found. If an ECA does not respond, it is considered a problem, and a corresponding FS is generated. For those SSP's receiving pseudo FS's, indicating only a change in state, the ECA must be accessed to retrieve the true equipment status for analysis. ECA's which have no alarm 'intelligence' are only capable of providing the status of their equipment and have no notion of what defines a correct as opposed to an incorrect state. Here the corresponding SSP uses the tables not only to poll the state of the equipment every few minutes, but also to define the relationship between the data received and the FS definitions.

The aim of the SSP is to arrive at a description of the problems concerning the equipment for which it is responsible, using FS descriptions. This result is stored in its AL and all changes are passed to the next software level, the LAS. All software concerning flat files, maintenance of the AL, acceptance of FS's from ECA's and passing the result to the LAS is contained in an alarm library. This, together with a SSP 'template' is offered to assist in building SSP's.

D. User Surveillance Programs

A USP offers users more independence to survey their system and often concerns equipment not connected to ECA's. This approach is used in computers dedicated to surveillance as

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S06SA03

well as in PCA's. Wherever it is used, the strategy is the same. Each USP surveys its system and arrives at a result which is transmitted as a SAV1 string to a corresponding SSP. Periodically, contact is made between the SSP and USP to align the AL of the SSP with that of the USP and to verify that the USP is still functioning.

E. Local Alarm Server

All FS's generated or dealt with in a computer will finally end up in the LAS, having passed via a SSP. The contents of the AL of a LAS represents the overall state of all equipment surveyed by that computer. This is in contrast to the contents of the AL of a SSP which only represents the state of the equipment monitored by that SSP. To ensure that the contents of the LAS AL equals the sum of the AL's of all SSP's in that computer, the LAS periodically demands the state of each SSP, correcting any discrepancies found.

It is the LAS which provides the external connection to the CAS, via a Remote Procedure Call (RPC) [6]. All changes of state in the LAS are immediately sent to the CAS.

There were 2 reasons for introducing the LAS level. The first was to reduce the number of physical connections to the CAS, as there were multiple SSP's at the level below. Secondly, it was to perform logical analysis on the FS's concerning all the equipment connected to that computer. For the moment no further analysis of FS's is performed at this level.

VII. THE CENTRAL ALARM SERVER

The CAS [7] represents the hub of the alarm system in that it receives FS's from all computers performing surveillance tasks via LAS's and distributes them to the various control centres for display to interested parties. Within the CAS there are a number of processes running, all communicating in a standard way using messages. These processes are concerned with: the management of the AL, which represents the state of the whole process; providing a 'backup' to each LAS to ensure AL consistency; access to the central alarm database which runs on-line within the CAS; archiving all FS's received; and finally dispatching relevant FS's to consoles.

It is the database which is the centre point for the processing within the CAS. All FS's known to the alarm system are present in the database. Any FS arriving from a LAS which is not known is placed in a trace file for further investigation.

When a FS arrives at the CAS it is in the form of a SAV1. The triplet FF/FM/FC is the key used to access the database, the remaining information in the SAV1 representing the dynamic part of the FS description. A number of relational tables are accessed and 2 types of static information are retrieved. One concerns FS details such as: person responsible, location address, installation concerned, in all, 10 fields. The second concerns information relating to who might be interested in the FS. This organisation is done within the database by grouping each FS into one or more 'categories', which represent areas of interest of the various users of the system. Examples of category definitions are: one for each equipment group, safety, machine operation, technical services etc. A match of a FS to one or more categories will return 6 fields of independent information for each matched category. This information concerns: description of the problem; action to be taken; priority, very serious, serious, and warning etc. All information is tailored to each category. For example a fire alarm attached to the safety category would have in its 'action' field: 'Immediate Intervention', since it is the safety services which deal with that type of problem, whereas the same FS associated with the machine operation category would be more for information or in some cases require the beam or equipment to be switched off depending on the location of the fire. Naturally FS's associated with machine operation would not be attached to the safety category.

As a result of this database access, a list is made of all the categories associated with the FS. It contains the SAV1 and both parts of the static information for each category. This represents all information known to the alarm system for that FS.

Any user who would like to receive alarm information at a console must run an initialisation which asks what categories of information are required. This operation informs the CAS and thereafter any FS received by the CAS which is associated with any of these categories will automatically be sent to that console. All communication between the CAS and consoles is made using RPC's.

VIII. RECEPTION OF ALARMS AT A GENERALISED CONSOLE

For the 'Injection Tests', a very simple, dedicated console was built using a 286 machine to receive FS information. It was soon found to be inadequate and it was decided to incorporate FS reception into the generalised console manager which was under development for workstations used in the control centres.

The Console Manager (CM), completely manages a workstation from 2 dedicated lines of 'icons' at the top of the screen, referred to as the CM banner. They provide information about LEP and allow the execution and control of multiple application programs, including the reception and display of errors encountered by these programs. To interface to the manager it was decided to dedicate an 'icon' within the CM banner and define it as a toggle with 2 states described by 'icon' texts: 'SHOW ALARMS' and 'HIDE ALARMS'. When a console is first initialised, only the CM banner is visible with the alarm 'icon' indicating 'SHOW ALARMS'. In this state no application programs are running.

To initialise the console to receive FS's the 'SHOW ALARM' 'icon' is selected which brings to the foreground 2 alarm windows and changes the 'icon' to 'HIDE ALARMS'. If the "HIDE ALARMS' 'icon' is selected, the CM removes the 2 alarm windows, but keeps them active in memory. One window has a single row of alarm 'icons' at the top and displays the AL; the other displays the instant faults. An alarm 'icon' called 'configuration' is used to configure the console for FS reception. This displays all the categories existing in the CAS. A selection of categories is then made which corresponds to those FS areas of interest. An 'apply' function then sends this request to the CAS, where a search is made for all AF's corresponding to that selection. All information concerning these FS's is then sent to the console. Thereafter any FS changes within these categories will automatically be sent to the console.

Alarm software within the console receives these states, places them in an AL, displays them in the active FS window, and manages a local archive. Information displayed is the date, FF, location address, FM, and fault description, all of which occupy 1 line. FS's are displayed in order of priority and time, with each priority having a different colour. The default is to display the most recent, highest priority FS's, but scrolling is possible.

Using the alarm 'icons', various operations can be performed on a displayed FS such as: display the dynamic and static information relating to that FS; acknowledge a FS by displaying it in inverse video; inhibit a FS, which sends a command to the appropriate PCA and SSP to flag and terminate that FS; etc. Another set of alarm 'icons' allows FS's to be re-enabled; the screen to be printed; scanning of the local archive; and creating a 'Test' alarm which checks most elements of the alarm system, finally arriving as a displayed FS in the active alarm window, and terminating automatically 20 seconds later.

Periodically the console alarm software requests a 'backup' to the CAS to verify its AL. If the console cannot access the CAS for any reason, it colours the alarm window blue and prints a message that the console has lost contact with the CAS. This is the final link in the chain which verifies the correct functioning of all parts of the alarm system from the point of generation of the FS in an ECA, right through to the display of that FS in the control centres.

To allow an operator to use the console to run other application programs, but at the same time be informed of any FS changes, the SHOW ALARMS 'icon' is used to indicate the arrival of any new FS.

Instant faults are not categorised. A console can either be initialised to receive all or none. The display is again 1 line per FS and works in 'roll over' mode with a scrolling facility.

Any console in the system can run the CM and initialise to receive FS's. This provides a very flexible method to connect to the alarm system. In practice we run with 6 permanent connections and up to 6 temporary ones.

IX. THE ALARM DATABASE

A. Overview

The database is the key to the overall system. Without it management of the system and interfacing to the equipment groups would not be possible. All FS's that can be generated, including all static information used to describe these states, are contained in the database. Relationships between FS's and categories, LEP states and equipment states are established. To enable all this information to be maintained, and at the same time use it in a coherent fashion, 2 relational databases with the same internal structure are used, one running on a centrally maintained VAX and the other in the CAS. It is the VAX database which is used for maintenance and interfacing to the equipment groups and is considered the 'master'. Archiving of all FS's arriving at the CAS is also maintained using both the database on the CAS and VAX.

B. The Master Alarm Database

Each equipment group interfaces with the alarm system through a standard flat table called the 'Interface Table' (IT), which was defined by the alarm team. It contains a complete description of all FS's generated by the equipment group. Roughly 90% of the column definitions are common for all equipment groups. The rest concern equipment specific information like data and FS relationships which differ widely. A set of scripts has been built by the alarm team which checks the consistency of IT's both with respect to themselves and the alarm database. The scripts must be run by the equipment groups before the IT's are used by the alarm system. Responsibility for these tables and the way they are interfaced to the equipment databases, lies with the equipment groups.

Preparation for a database update consists of: verifying any changes made to IT's; using these tables to generate all necessary flat files for those SP's affected by the changes; and updating the master database from the IT's. At this point the CAS is stopped, the database loaded, and restarted. All affected SP's are also stopped, internal tables initialised from flat files, and restarted. This operation takes about 15 minutes.

C. The CAS Database

The database running in the CAS is used only in 'select' mode. For each FS arriving, the database is accessed to: check its existence in the database; append all static information known about the FS; and classify the FS according to the user 'categories' of interest.

D. The Central Archive

Each FS that arrives at the CAS is archived. The archived information consists of the elements of the SAV1 string which is stored in a flat table. This means that to complete the description of a FS in the archive, access must be made to the alarm database. To avoid the management problem of storing archived information, a minimum archive is stored on the CAS. Each day, the previous day's archive is automatically transferred to the VAX, and removed from the CAS. The VAX manages 2 alternating archive tables which are 'record' limited. When the current table reaches its limit, it is copied to a file and the other table becomes the current archive. In this way a continuous archive is available on-line. Facilities to access this archive from consoles initialised to receive alarms are available.

X. SPECIAL REQUIREMENTS FOR SAFETY

In order to satisfy the safety requirements, 3 main areas had to be considered. First, all FS's associated with the safety of personnel had to be categorised according to the action required. These actions were: immediate intervention by the safety service, approximately 500 FS's have been defined for this category; immediate intervention by the technical service; and intervention by the technical service during working hours.

Secondly, the transmission of FS's to the SCR had to consist of 2 independent paths: one using the control system to transmit details; and the other, called the 'redundant' channel consisting of 1 summary state per LEP point transmitted via a 'hardwired' line to a synoptic panel in the SCR.

Thirdly, all the active components in the computer and hardwired transmission paths to the SCR had to be powered by secure electrical power. To reduce this equipment to a minimum, 2 computer networks around LEP were installed: one for the machine and one for safety including the services.

XI. EXPERIENCE WITH NEW TECHNIQUES

As a continuation of our efforts to explore the possible uses of ES's within the LEP alarm system, a pilot project [8] was launched to build a prototype surveillance system for the power converters of LEP. This work followed a thorough investigation of the commercial market for products which were suitable for our environment, and which could do as well, hopefully better, than our running conventional system.

We were looking for a modular, 'real time' ES running in a UNIX environment with the ability to link to standard commercial software. The 'real time' aspect was not so much the response time, although that was important, but rather the possibility to reason over time using temporal constructs. Modularity was important because we already had equipment data acquisition, networking, etc.

Finally we found a system which seemed to have all desired features, as well as an interface to a graphical package which we were already using. It was an American product and perhaps because it had no established European agents, we found it impossible to get the necessary technical support to allow us to continue with the product.

This was a major set back, but we decided to visit, for the second time, a commercial exhibition of ES's in France. A European product which seemed ideal was found. Training was arranged for 2 of our personnel, and a prototype prepared for the LEP power converters to be used in an evaluation of the product for one month. The first month was spent in trying to load the product, and thus little work was done on the prototype. An extension of 1 month was arranged for the evaluation and when work finally started on our prototype it soon became clear that the product was not operational and, apart from numerous bugs, a number of the advertised facilities either did not work as described or did not work at all. Subsequently we found that the product was no longer on the market. This concluded our work in this area to-date.

XII. CONCLUSION

The LEP alarm system today provides an important, reliable facility for machine operations, and for technical and safety services. It has shown itself to be flexible to equipment changes and capable of accepting upgrades gracefully. The components from which it is built are well interfaced and can be independently changed. Capacity of accepting new FS's has been demonstrated recently by connecting the complete technical services of the Meyrin Site, some 5000 states, representing 6% of the current CAS total.

The decision to pursue the conventional approach was correct and there remains doubt that the system would have been ready in time if the alternative approach had been taken.

XIII. ACKNOWLEDGEMENTS

I would like to thank the many people who have contributed to this project, especially: Tuanhua. Huang*, B.Marchal*, and L.Mantegazza for the database; J.M.David*, F.Rodor*, and J.M.Ortholand for the CAS and MMI; J.P.Forestier* for the PCA; and C.A.Cameron for research into new techniques and continued development on the PCA.

* people no longer at CERN.

References:

[1] J.Fuchs,P.Skarek,L.Varga,E.Wildner-Malandain,Integration of Generalised KB-Sytems in Process Control and Diagnosis, 'SEAS' Conference, Lausanne, Switzerland, April 8-13, 1991.
[2] F.Chevrier, Safety Controlled By An Expert System On Experimental Sites In High Energy Physics, Pergamon Press, Operational Expert Systems Applications in Europe, 1991.
[3] J.Ancelin,J.P.Gaussot,P.Legaud, A Real Time Knowledge-Based Alarm System EXTRA, AI & other Innovative Computer Applications in the Nuclear Industry, 1987 Snowbird Utah, Plenum Press, New York & London.
[4] P.G.Innocenti, The LEP Control System, Conference on Accelerator & Large Experimental Physics Control Systems, Vancouver, BC, Canada, October 1989.
[5] J.P.Forestier, The PCA Alarm Software Environment & Surveillance Software, CERN/SL/CO/Note/91-15 (unpublished).
[6] P.S.Anderssen,V.Frammery,Tools for Remote Computing in Accelerator Control,Conference on Accelerator & Large Experimental Physics Control Systems, Vancouver, BC, Canada, October 1989.
[7] F.Rodor,The LEP Central Alarm Server & Console Interface,CERN/SL/CO/Note/91-7/8,(unpublished).
[8] C.A.Cameron,LEP Alarm Group Expert System Project Status Report,CERN/SL/CO/90-122,(unpublished).

Trademarks:
UNIX is a trademark of AT&T Bell Laboratories.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S06SA04

The Software for the CERN LEP Beam Orbit Measurement System

Giulio Morpurgo
CERN/SL, 1211 GENEVE 23, Switzerland

The Beam Orbit Measurement (BOM) system of LEP consists of 504 pickups, distributed all around the accelerator, that are capable of measuring the positions of the two beams. Their activity has to be synchronized, and the data produced by them have to be collected together, for example to form a "closed orbit measurement" or a "trajectory measurement". On the user side, several clients can access simultaneously the results from this instrument. An automatic acquisition mode, and an "on request" one, can run in parallel. This results in a very flexible and powerful system.

The functionality of the BOM system is fully described, as well as the structure of the software processes which constitute the system, and their interconnections. Problems solved during the implementation are emphasized.

Introduction

The Beam Orbit Measurement (BOM) [1, 2] system is one of the most vital instruments of LEP, and it is potentially very powerful. 504 pickups, connected to 40 VME crates, are distributed all around the LEP ring . Each pickup can acquire a signal induced by the passage of each single bunch of particles and record it in a VME compatible memory card. Each crate is equipped with two such cards, ("Main" and "Secondary"), which can store signals coming from up to 1800 (450 for the secondary) turns. The signals are processed locally by microprocessors in the VME crates (called DSC, from Device Stub Controller), and then finally collected in a single computer to produce the final result (typically, the LEP orbit).

Due to various reasons (limited CPU power in the VME crates, incompleteness of the software, unfriendliness of the RMS68K programming environment on the VME crates and the reorganisation of the CERN accelerator divisions), the potential of the system had never been fully exploited. For this reason, in July 1990 it was decided to upgrade the computer part of the BOM system, both from the hardware and from the software side. Major points of the upgrading were :

1. Replacing the DSC 68010 CPUs with 68030 CPUs equipped with floating point coprocessor.
2. Replacing the RMS68K microprocessor operating system of the DSCs with OS9. The new DSC systems would be diskless and would load system and application programs from a common file server. This would make the overall system cheaper, more reliable and easier to maintain.
3. Using a dedicated Apollo workstation (the BOM Server) to collect the data coming from the DSCs and to deliver the results to the users of the BOM system.
4. Implementing direct network connections, based on the TCP/IP protocol, between the DSCs and the Apollo workstation where the data have to be put together.
5. Finally, a complete redesign and rewriting of the software to be run both on the DSCs and on the Bom Server . This will be the main subject of this article.

Constraints (and flexibility) : How the system works.

When upgrading an existing system, some degrees of freedom are frozen. A preliminary step to perform is to examine the things that cannot be changed; the new system will have to live with them.

In the LEP BOM system, the data acquisition in the 40 DSCs is triggered and synchronized by the Beam Synchronous Timing (BST) system [3, 4, 5]. This system interprets "tasks" written in a pseudo-assembler language. At each LEP turn the BST distributes an identical message to all 40 DSCs. This message contains a part which can be read by programs running in the DSC, and a part directly received by the hardware installed in the VME crate. As mentioned in the Introduction, in each crate there are two "Acquisition" memory cards. By software it is possible to independently set the access to these memories in one of two modes ,"intern" or "extern". In the "intern" mode the memory is made accessible to the processes running on the DSC CPU, whereas in "extern" mode the memory can receive data from the BOM data acquisition electronics (FADCs). The FADCs produce data every time a bunch of particles crosses the pickup. If the access to a memory is set as "extern", and a certain bit of the BST message received at a given LEP turn is set to 1, then the data produced by the FADCs during that turn will be written into the memory, together with the arrival times of the various bunches.

The pickups close to the intersection points are equipped with Wide Band electronics, while the rest of the pickups are equipped with Narrow Band electronics. The Narrow Band pickups can measure, during the same turn, the signal generated by each bunch of each beam circulating in LEP. The Wide Band pickups only measure the signal generated by bunches of a preselected beam.

The fact that the system has two acquisition memories provides a certain flexibility. It is able to perform two different operations at the same time, one in the Main memory, the other in the Secondary memory. The two memories can be considered as belonging to two different instruments.

There are, however, a few constraints which have to be dealt with :

1. While the Narrow Band systems can acquire both beams at the same time, the Wide Band systems cannot. Their settings have to be changed if one needs to change the type

of beam to be observed.

2. The hardware setting of the pickups is common for all the pickups in a given DSC and for both acquisition memories.
3. The BST message received at each LEP beam turn will always be the same for all the DSCs.
4. Two other possible limiting factors are the data processing capacity of the BOM CPUs and the data transfer capacity of the network connecting the BOM crates to the Apollo BOM Server.

Requirements : what the BOM should do

The starting point of the new BOM software design was the analysis of the requirements. This implies the definition of the operations the system is expected to perform, and of what is important to the users who perform the different operations.
Several different operations can be performed by the BOM System, by acquiring different sets of turns and by extracting different information from the acquired data. We briefly describe each of them.

- **Closed Orbit Measurement.** "Acquire a given number of turns and, for each pickup, compute the average beam position. "
  The user must be able to specify the number of turns, and to tell the system if data coming from the two beams have to be mixed or not.
  The average is made over all bunches of a given beam, or over all bunches of both beams if the result has to be mixed. This is the most important of the BOM operations, because it is used for correcting the orbit. It is therefore very important that the BOM System should be able to perform it quickly and reliably.
  This operation requires synchronization between the 40 BOM crates : we need to acquire data from the same turns all around LEP. Therefore the start of the data acquisition must be triggered by the BST.
- **Beam Trajectory Measurement.** "Select a beam. Acquire a given number of turns. For each pickup compute the beam position of each bunch at each turn. "
  Again, this operation requires absolute synchronization between the 40 BOM crates. In this case it is particularly important that every pickup acquires data from the same turns
  The Trajectory Measurement becomes very important when, for one reason or another, the beam does not survive inside LEP. In this case the acquisition will be triggered at the injection of particles into LEP, so that the behaviour of the beam immediately after the injection can be analysed.
  Typically one is interested in a very few number of turns (8 turns are enough to get the first revolution for each bunch).
- **Beam Position Monitoring at Experiments.** "Every 60 seconds, acquire and average the beam position in the pickups closest to the intersection points with experiments (Wide Bands), and derive the beam position at the interaction points"
  This operation is only important when LEP is running in Physics mode, and it has to be performed only on the Wide Band crates.
  One can note here that if the BOM system periodically measured the Closed Orbit, the same data could also be used to perform this measurement.
- **Continuous Recording** (for Post Mortem Beam History or other) "Every n turns, acquire 1 (or m) turn(s). On request, or on a special event, stop the acquisition." The aim of this operation mode is to be able to reconstruct the behaviour of the beams before a given event (e.g. beam loss). The data processing will depend on what the user is looking for.
  One should note that, since for the Wide Band only one beam can be acquired at a time, the information coming from those pickups will be incomplete. Note also that unless an automatic mechanism like the BST "service request" is available to stop the acquisition in real time, this operation will not be really useful. In fact, if the acquisition takes place every turn, the memory will be completely overwritten every tenth of a second.
  This mode of operation is particularly important during LEP Machine Development.
- **1000 Turns Acquisition.** (for Harmonic Analysis of individual Pickup Data, or other). "Select a beam and a particular bunch. Fill the memory with consecutive turns, for each pickup compute the positions of the selected beam and bunch at the different turns, and store them in a table. "
  The table is then available for whatever analysis is required on the data. It must be possible to start the acquisition after a specified event.
  It could also be convenient to be able to acquire a turn every n turns, to observe the beam over a longer time interval. In fact, 1000 consecutive turns represent only 89 milliseconds.
- **Calibration and Simulated orbit.**
  "Calibration : With no beam in LEP, use the ad-hoc BOM hardware to determine the gains and the offsets for each pickup. Save these values, and detect suspicious pickups."
  "Simulated Orbit : With no beam in LEP, use the ad-hoc BOM hardware to simulate an orbit with predetermined positions. Check if the measured values match the predicted ones."
  The Calibration was normally used by the BOM hardware specialists, but making it more automatic, it could become a routine procedure for the operators. The Simulated Orbit is used to check the quality of the Calibration.

Some of the operations described are performed on demand, others are executed continuously in background, others are repeated at fixed intervals. The most important operation is the Closed Orbit measurement, which is needed to correct the orbit.

The strategy of the new software

All the described BOM operations consist of two main parts : the *acquisition* and the *processing of the data*. The acquisition includes also the preparation of the system (hardware setting). The data processing includes also, in most cases, sending the data to the BOM Server, which has to collect the data from all

the DSCs and to make them available to the users of the system. To synthesise our analysis, we have to find a convenient solution to exploit the flexibility of the system (two independent memories in the DSCs) in order to satisfy the requirements (on demand and automatic operations), keeping in mind the overall constraints (only one hardware setting , and only one BST task executable at a time, that means only one *data acquisition* operation performable at a time).
Solving this problem in terms of software means finding a natural subdivision of tasks between different computer processes, both on the DSCs and on the BOM Server.

Processes on the DSCs

On the DSCs we have 4 main processes : one to *Prepare* logically the DSC for the following operations, one to *Set the Hardware* of the DSC to the required conditions, one to *Process the Main Memory* and one to *Process the Secondary Memory*. These two latter Data Processing processes are two instances of the same one, accessing respectively the Primary and the Secondary Memory. All these processes wait for BST messages, and then perform the specified actions. Typically any of the BOM operations will consist of a call to the Prepare process, one or more calls to Hardware Setting followed by acquisition of data in one of the two BOM memories, and eventually a call to the corresponding Data Processing Process, which will read the data from its memory, will produce the result and will send it to the BOM Server if needed.
Other two processes perform useful tasks : the *Auxiliary* process and the *Creator*. The Auxiliary process performs all sort of miscellaneous operations (setting the DSC time, loading new calibration factors, doing offline analysis on data already acquired) on reception of the corresponding BST action code. In particular, it performs the harmonic analysis on the table produced by the Main Memory process as a result of the Multiturn Acquisition. The Creator process starts all the others, restarts them if they die, and creates the shared memory areas via which the other processes can share information.
An additional independent process is an RPC [8] server for diagnostic access to the hardware of each BOM crate from PCs and Apollos (The RPC is used in this case because the Xenic PCs found close to the equipment do not support TCP/IP sockets).

Processes on the Apollo BOM Server

On the Apollo BOM Server two processes , the *Automatic* server and the *Demand* server, share the BST resource via a semaphore. While the Automatic server executes periodically commands from a command table, the Demand server is activated by the Interface server. This latter process constitutes the interface between the BOM Server system and the users. Via a library of functions they can, among other things, trigger a new closed orbit measurement, ask for the result of the latest or the next acquired orbit, enable or disable the automatic server. Two other processes, the Collector and the Receiver, complete the building blocks of the BOM Server. The Receiver receives data from all the DSCs directly into a shared memory, from where the Collector can read and treat them. The Collector is also signalled by the Demand or by the Automatic servers every time an operation (e.g. Closed Orbit) starts, and tells the Interface every time the results of an operation are completed. The Interface will then deliver the result to the user(s) waiting for it, if any. All these processes are started by a Starter process, which surveys them, and communicate through shared memories (MOPS) and message queues.

Implementation Problems

The implementation of all these ideas required a considerable amount of work. We had to find our way through several foreseen and unforeseen problems. Some have been solved satisfactorily, others not yet.

- SPEED.
  In order to improve the performance of the DSC software, special care has been taken to avoid repeated calls to subroutines with high number of arguments. Also "register" variables have been used in an effective way. By following these prescriptions, it was possible to save an order of magnitude in the DSC processing time.
- PARTICLE IDENTIFICATION.
  The Narrow Band Pickups acquire signals coming from each bunch of each particle. The software analysing these signals should be able to distinguish between electrons and positrons. For this reason, the relative arrival time of each bunch with respect to the LEP turn clock signal is recorded in memory together with the data. This time (called *"fine-time"*) is measured in units of 400 nanoseconds, and it can be compared with reference tables for each pickup, to identify the different bunches and to reject parasitic signals. When all the bunches are present, the particle identification is easy, because the arrival order of the bunches if predetermined. When some bunch is missing, the identification becomes critical for the Narrow Band pickups closer to the intersection points, because the time interval between bunches of the two particles is only around 660 nanoseconds, and the turn clock signal arrival time can be wrong by up to 200 nanoseconds. To try to solve this problem, we first analyse the data for a pickup far from the intersection point. From this, we produce a table containing the different bunches in LEP, together with the difference between the theoretical arrival time and the measured one. This table enables the program to correctly identify the particles even in the most critical pickups.
- INSTALLING THE NEW SYSTEM.
  The transition between the old and the new system was not a single step one. It had to be done progressively, keeping the whole system operational, and replacing old parts with new ones as soon as they were ready. Therefore it has been necessary to make all the new software produce results compatible with the old one. This has been achieved by writing a process (the *BOM Listener*) on the Apollo BOM Server. The BOM Listener receives from the Collector all the messages that should have gone to the Old Collector on the PC, and sends these messages to it. Another point where compatibility with the old system was taken into account was in writing the new BST tasks, because they had to be understood by both systems.
- NETWORK TROUBLE.
  The most serious problem was the network communication between the DSCs and the BOM Server Receiver. We are currently using the TCP/IP communication protocol, connecting each DSC to a Receiver program on the Apollo.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S06SA04

We do not have a large amount of data to send (typically less than 500 bytes per DSC), but we meet serious limitations. To make a long story short, we need to run 8 Receivers on the Apollo, so that they will receive data from 5 DSCs each. We found that if we tried to send all the data packets from the DSCs at the same time, a large fraction of the packets were arriving with large regular delays (typically 13, 26, 40 and 92 seconds), and occasionally some packets were lost. The regularity of these time intervals implies that we were observing some "wait and retry" mechanism implemented inside the TCP/IP software. In our set up this effect started to manifest itself when more than 15 DSCs were installed. Our first attempt to circumvent this problem has been to introduce additional delays in the DSCs, so that they do not try to transmit all at the same time. Our best result has been achieved by spreading the transmission of the DSC packets over around 10 seconds, in which case most arrive in a spread of 12 secs, with only a few arriving after around 25 secs.

To improve this performance, an alternative data transmission has been developed, based on a simple protocol sending raw Ethernet packets from the 40 DSCs to a central collector OS-9 system on the same Token Ring. A single TCP/IP connection then connects this system to the BOM Server Apollo. This system seems to reliably deliver all 40 replies within at most 12-13 seconds, and often quicker (the best observed performance was 7 seconds) ; we are gaining operational experience with it.

- INVOKING THE BST SYSTEM.

  The BST system is currently activated through a remote procedure call to a dedicated PC. Up to a few seconds are lost in establishing the connection to this system. Moreover, occasionally the calling program never returns from the RPC routines, and it has to be killed.

  We are currently porting the BST system to the OS9 environment.

- NETWORK FILE SYSTEM FOR THE DSC.

  Our DSCs are diskless and boot from a centralized fileserver. Initially, a reduced version of the Network Disk server developed in the LEP Aleph experiment [6] was developed by D. Mathieson to run on OS-9. This support work was essential, because it allowed tests with a full 40 DSC system to be carried out during the 90-91 winter shutdown of LEP, before a usable NFS was available for OS-9. This system was also used in LEP operations until June, when the NFS on OS-9 seemed sufficiently stable to transfer to using it.

  This latter system is far from being perfect. In order not to overload the fileserver and the network, we have to wait around 10 seconds between the boot of two DSCs. With 40 DSCs, this means around 7 minutes to boot all of them; yet not all successfully reboot at the first attempt. A more serious problem is that the online use of the system is almost impossible; if a few DSCs want to access files at the same moment, very often the NFS gets stuck and it is unavailable for about 30 minutes. A better configured fileserver, closer to our DSCs and more dedicated to the BOM , is planned to be installed.

Results

The transition from the old to the new system has taken place in several steps. First, in mid April 1991, the first version of the Apollo BOM Server was installed, together with the first OS9 Wide Band DSC. At the beginning of May the remaining 7 Wide Band DSCs were installed, and since this time data from the Wide Band systems has been sent to the Apollo, which forwarded it to the old BOM Collector PC. The next step was to use the first Automatic Server to periodically produce the data needed to determine the position of the two beams at the interaction points [7]. This was accomplished in May. Also in mid June the 16 Narrow Band DSCs which share the same network infrastructure with the Wide Band DSCs were installed during one afternoon. At the beginning of July, profiting from the last access hours during a short shutdown, the remaining 16 NB DSCs were rapidly installed and connected to the LEP Token Ring, using the freshly arrived IBM Ethernet-Token Ring Bridges. In August the new Multiturn Acquisition and Harmonic Analysis facility was first demonstrated. After the holidays, work went into improving the Calibration procedure and producing a second version of the BOM Servers. Currently the Automatic Orbit Acquisition is running, producing a new orbit every 60 seconds. The latest orbit produced is available to the LEP operators within seconds. Applications based on the repeated orbit acquisition have been written, for example to monitor and display continuously the position of the beams at all the Wide Band Pickups in the Experimental Areas.

To summarise the results

- Thanks to higher performance DSC CPUs and to software better optimised for speed the data processing time has been reduced by a factor of 25 . This means that it is no longer impractical to average the Closed Orbit Measurement over a large number of turns. This turned out to be very useful when we discovered, via a Multiturn Analysis, that there was strong 50 Herts noise in LEP (generating a typical peak to peak oscillation of 0.5 millimeters). The period of this oscillation is 224 turns, so, in order to average it out, we now measure the orbit over 224 or 448 turns. The processing time takes only a very few seconds. The overall time from when a new orbit measurement over 224 turns is asked to when the result is made available to the operators is now around 22 seconds (using the above mentioned alternative data transmission schema). We plan to reduce it to 15 seconds next year.

- A new algorithm has been implemented, with good results, to distinguish between electrons and positrons in the Narrow Band Systems. This makes it possible to acquire an orbit of a specified beam. It was possible in this way to show the saw_tooth effect due to the beam losing energy when emitting synchrotron radiation.

- The Multiturn Acquisition, together with the Harmonic Analysis facility, has been used successfully to measure some physics parameters of LEP (phase advance, dispersion,..). The effect on the injection kick on the orbit can also be examined, and instability in the beam detected and studied. A Fast Fourier Transformation can be also performed on each pickup's data , to show possible perturbations.

  By analysing data from the Multiturn Acquisition it was possible to localize the source of the 50 Herts noise.

  This facility also enables us to check the quality of the data coming from the different pickups, and makes the detection of bad pickups easier. By examining results from such an acquisition , we were able to detect a pickup which was wrongly cabled (the vertical and horizontal positions were interchanged), and another where two cables were short-circuited.

- The Automatic Orbit Acquisition, by which a new orbit measurement is produced every minute, makes life easier for the operators. It also constitutes the base for any LEP orbit statistics program. By a simple call to a function, any program can get all the data coming from the latest acquired orbit within seconds.
- The Calibration, which in the past was not very user friendly and required half a day of work, now takes 10 minutes and does not require any specialist intervention. It also updates a database, through which pickups which were not correctly calibrated will be marked as bad in the orbit data provided to the Operators.

Future Improvements

Of all the initial requirements, only the Continuous Acquisition of data has not yet been implemented.

Acknowledgements

Despite the fact that the BOM software described in this paper has been written essentially by one person, the project owes its success to the fruitful collaboration of many people. In particular, I would like to thank very much my colleagues : J.Borer who had the necessary patience for explaining the BOM system to me, C.Bovet, A.J.Burns, D.Cocq and A.Manarin for their precious collaboration, J.J.Gras for the BST receiver server and driver used in the BOM DSCs, C.David for improvements to the BST Master software, and D.Mathieson, who contributed in an essential way to the project by providing the network disk system used until June, by designing and implementing the raw Ethernet communication software mentioned above, and by generally supporting the required development environment. In addition, although independent from the software described in this paper, H. Michel has provided the specialist BOM diagnostic programs.

References

[1] J. Borer et al. *The LEP Beam Orbit Measurement system.* Proceedings of the 1987 I.E.E.E. Particle Accelerator Conference, Washington D.C, 1987.

[2] G. Baribaud et al. *The LEP Beam Orbit Measurement system : status and Running-in results.* Proceedings of EPAC, Nice, 1990.

[3] G. Baribaud et al. *The Beam Synchronous Timing system for the LEP instrumentation.* Proceedings of ICALEPS, Vancouver, 1989.

[4] C. David. *The Beam Synchronous Timing system for the LEP instrumentation. The Master Reference Manual : new facilities.* CERN/SL/BI-SW/Note 91-19

[5] JJ. Gras. *The Beam Synchronous Timing Receiver Utilities.* CERN/SL/BI-SW/Note 91-01.

[6] T. Charity *OS-9 Network Disk Server, User Guide and Reference Manual.* CERN/EF/ALEPH 89-194.

[7] A. Belk. *Interaction Point xx' Software for LEP. IPoptics and IPserv.* CERN/SL/CO 91-29

[8] P.S.Anderssen et al. *The Network Compiler User Guide.* CERN LEP Controls Note 97

[9] W. Herr *M.O.P.S. User Guide for "C" Programs.* CERN/SPS/88-43(AMS)

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S06SA05

A VMEbus General-Purpose Data Acquisition System

A. Ninane, M. Nemry, J.L. Martou and F. Somers
Institut de Physique Nucléaire, Université Catholique de Louvain
Ch. du Cyclotron, 2 – B-1348 Louvain-la-Neuve, Belgium

November 8, 1991

Abstract **— We present a general-purpose, VMEbus based, multiprocessor data acquisition and monitoring system. Events, handled by a master CPU, are kept at the disposal of data storage and monitoring processes which can run on distinct processors. They access either the complete set of data or a fraction of them, minimizing the acquisition dead-time. The system is built with the VxWorks 5.0 real time kernel to which we have added device drivers for data acquisition and monitoring.**

The acquisition is controlled and the data are displayed on a workstation. The user interface is written in C++ and re-uses the classes of the Interviews and the NIH libraries. The communication between the control workstation and the VMEbus processors is made through SUN RPCs on an Ethernet link.

The system will be used for, CAMAC based, data acquisition for nuclear physics experiments as well as for the VXI data taking with the 4π configuration (100 neutron detectors) of the Brussels-Caen-Louvain-Strasbourg DEMON collaboration.

I. INTRODUCTION

Experiments differ in the way they produce data: they use different standards of hardware to digitize data (`VME`, `VXI`, `CAMAC`, ...); they generate data varying in byte length and counting rate. However, the last stages of data acquisition systems have many things in common: the data are analyzed on-line to control the experiment and are written on storage devices for further off-line analysis.

We have defined a common framework for a general-purpose data acquisition system. It meets the following requirements:

- the data source is open: the system can be enabled to acquire data from various instrumentation buses;
- the data sink is open: data can be analyzed on-line by concurrent processes and can be stored on different types of mass storage devices;
- the system is scalable: it can be used for low count rate nuclear physics experiments (20 byte events at 200 Hz) as well as in larger experiments such as the 100 neutron detectors of the `DEMON` collaboration [1] (300 byte events at 5 kHZ);
- the user sits at the highest level of the data acquisition system with the modern conveniences of workstations.

II. SYSTEM ARCHITECTURE

A. Distributed Hardware

The system is designed following a distributed architecture (Figure 1). The real-time data acquisition is performed by a `VMEbus` system. It allows to connect a wide variety of interfaces to external hardware as well as to run data acquisition processes by various processor boards. The user acquisition control and data handling is delegated to a standard workstation connected to the `VMEbus` system by an Ethernet link.

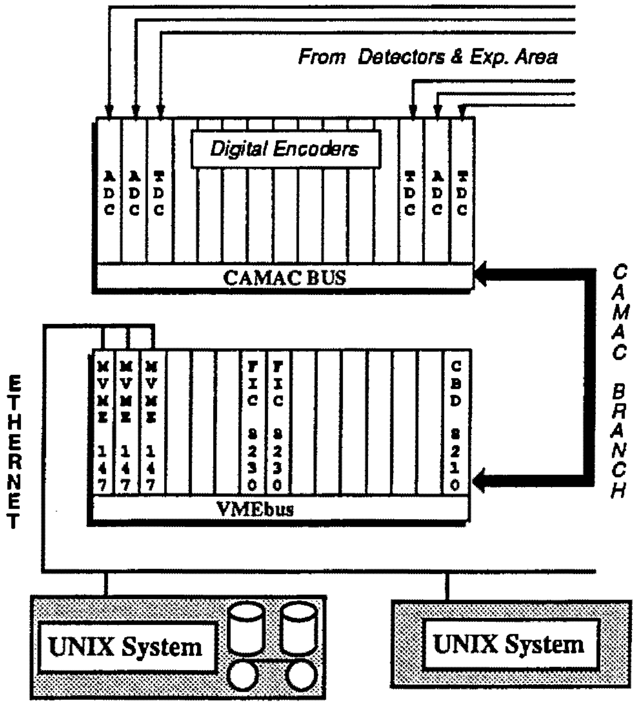


Figure 1: A Simple Distributed Architecture

In such an architecture, both parts are loosely coupled and may be evolved on their own. The user workstation or the `VMEbus` system may be replaced or upgraded without redesigning the entire system.

B. Software Architecture

Events are defined as a set of structured and correlated data which enter the system at random by interrupts. Events[1] are assembled into larger, configurable, structures called *blocks*.

A number of tasks can be implemented in the system to read data simultaneously from data *channels*. A channel is characterized by an access mode: `full` or `sample`. Tasks accessing data through a `full`-mode channel read and process all the data blocks. They can therefore lead to a considerable increase in the system dead-time. The influence on the data processing dead-time can be reduced by `sample`-mode channels which access only a sample of the data at the task's own processing speed. A data storing process works in the `full`-mode, while the `sample`-mode suffices for data monitoring. A `block type` parameter can also be assigned to a data channel: read operation on the channel will return only blocks of events of this particular type.

The Buffer System

The data acquisition system can be viewed as a *producer* task – the event's interrupts – and many *consumer* tasks – the data analysis and storage – running concurrently to fill and consume blocks of events. The producer and the consumer tasks share a common buffer system.

A *buffer* refers to a block of events. They are arranged in two doubly-linked lists [2] (Figure 2): the *free* list contains buffers that can be used directly by acquisition interrupts to store new events, while the *valid* list contains buffers already filled with events but not yet processed. Buffers can reside in both free and valid lists. This situation occurs when they have been processed by all `full`-mode channels but not by all `sample`-mode channels.

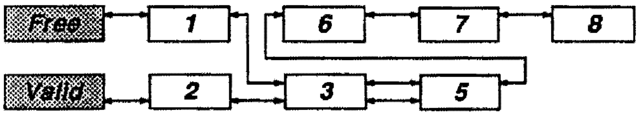


Figure 2: The two doubly-linked lists

Producer Part

At the beginning of a data acquisition, all buffers reside in the free list. When the data acquisition process starts, a buffer is extracted from the head of the free list and becomes the *current buffer*. It is filled with events up to its maximum size. It inherits the identification of the data reading channels interested to process it and is added at the tail of the valid and/or the free lists. A new current buffer is extracted from the free list head and the procedure continues.

Consumer Part

A data reading channel scans the valid list until it finds a new unprocessed buffer and returns the data to its parent task. When the operation is completed, the buffer is marked and is moved within the linked lists according to three situations:

1. the buffer waits to be processed on another `full`-mode channel: nothing happens. It remains on the valid list and is safe from interrupts;
2. the buffer waits to be processed on `sample`-mode channels only: it remains on the valid list and returns at the tail of the free list;
3. the buffer has been processed on every channel: it is removed from the valid list and returned to the free list.

If the acquisition produces data at a faster speed than the consumers process them, the free list will be emptied; event interrupts are then disabled until a consumer process returns a buffer to the free list.

III. IMPLEMENTATION

The ideas presented above have been implemented in a `VMEbus` system running the `VxWorks 5.0 kernel`.

A. Hardware

The `VMEbus` system consists of three `Motorola MVME147` boards with `MC68030` microprocessors. Each board has `SCSI` and `Ethernet` capabilities although they are not used on all of the boards. The system has been used so far with two different sources of data: `CAMAC` and `FIC8230` preprocessor.

CAMAC

The `CAMAC` crate is connected to the `VMEbus` by the `CES CBD8210` branch driver and the `CCA2` crate controller. The module allows the generation of `CAMAC CNAF` cycles as `VMEbus` memory mapped addresses. This elegant feature provides a fast access to the `CAMAC` bus and facilitates the software writing. The data acquisition system is interrupted at each physical event by `CAMAC LAMs`.

FIC Preprocessor

The `CES FIC8230` is a `VMEbus` board with a `MC68020` microprocessor. It runs a fast, specifically developed, kernel. The processor receives events from a `CAMAC` crate or from a `DMA` channel connected to local hardware. The microprocessor assembles events into blocks which are then written directly to the last stage of the data acquisition in a single interrupt.

B. Acquisition Software

To reach a high level of flexibility, the data acquisition system has been layered (Figure 3). The real-time kernel executes user tasks, which control the acquisition process and read the data through the kernel `I/O` system. At a

---

[1] The event definition is not restrictive: an event can be `CAMAC` data of a single physical event but can also be a block of data preprocessed or filtered by other processors.

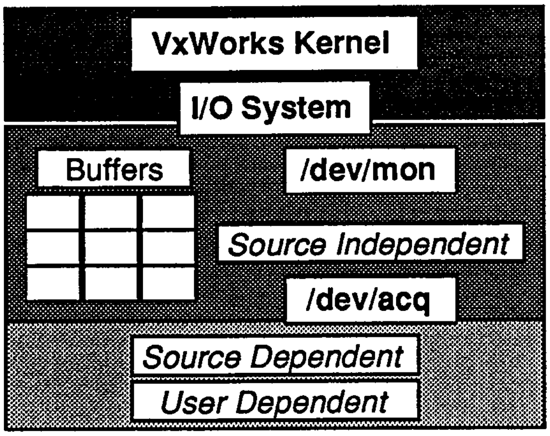


Figure 3: Acquisition System Layering

lower level, the data acquisition software itself has been structured in three layers:

1. the *source independent* layer comprises the data buffer system and its integration into the kernel I/O system;
2. the *source dependent* layer implements routines to connect the data source to the acquisition system;
3. the *user dependent* layer implements routines to render the acquisition process suitable to the user needs.

Real-Time Kernel

The VMEbus processors run the VxWorks 5.0 real-time kernel. This software has been selected for its:

- clear separation between system development and code management tasks – on a UNIX system – and real-time tasks – on VMEbus processor boards – which fits naturally in our distributed architecture;
- platform independency on both sides: many UNIX platforms and many VMEbus processors are supported;
- high networking capabilities with IP family of standard protocols (TCP, UDP, RPC, NFS, ...).

The kernel provides all real-time primitives: semaphores, events, message queues, control of preemption, priority-based scheduling, ... as well as the standard C library. It supports the notion of *device driver* which provides a common interface to devices or pseudo-devices through calls to the standard I/O C library.

Multiprocessor Extension

The VxWorks 5.0 is basically a single-processor kernel. To use the full power of the VMEbus system and to obtain the requested scalability, multiprocessor (MP) features have been added. The granularity of the MP-architecture is situated at the task level.

The system has a *master* processor and many *slave* processors. The master creates and owns the shared ressources while the slaves manipulate them. We have implemented MP-devices and MP-semaphores.

MP-devices: Device structure has been splitted into a private and a shared part. The private part is the standard VxWorks device structure refered to the local I/O system. The local structure contains a reference to the shared part of the device.

MP-semaphore: The MP-semaphore has been implemented with a shared flag protected by a *spinlock* variable [3]. The spinlock is accessed by indivisible cycle machine instructions to eliminate contentions. The MP-semaphore has a private, standard VxWorks, semaphore in each of the participating processors. Tasks waiting for the MP-semaphore *sleep* on the private semaphore inside their processor. A remote *wakeup* has been implemented with the help of the MVME147 mailboxes.

System drivers

The data buffer system is accessed by two MP-device drivers integrated in the VxWorks I/O system. They reflect the producer–consumer relationship.

1. /dev/acq: The acquisition device controls the *production* of the data. ioctls are used for example to start and stop the acquisition by enabling and disabling the interrupts in the master board.
2. /dev/mon: The monitoring device implements the *access* to the data. Tasks open this device to get a channel and read data.

The lower part of the acquisition device driver is connected to the data source by four routines:

1. acqStart(): implements commands to initialize the source when starting an acquisition process;
2. acqIntr(): is executed at each event interrupt;
3. acqRestart(): restarts the data source at the end of the event interrupt handling;
4. acqStop(): executes commands to finish the data acquisition process.

Each routine has an user defined part, which accesses the user modules participating in the data acquisition.

Because of the VMEbus limitation of a single interrupt handler on a given level, the data acquisition process can be executed only on a single processor, the master, while the data processing tasks run on several slave processors.

Network Servers

The remote control and data analysis from an user workstation is executed by *Remote Procedure Calls* (RPCs) *servers* running in the VMEbus system.

- acqServer: executes ioctls on the /dev/acq device to control the data taking;
- monServer: controls the /dev/mon device to grant access to channels for remote data reading tasks;
- acqSysServer: supervises global parameters and procedures such as system directory, system reboot, ...

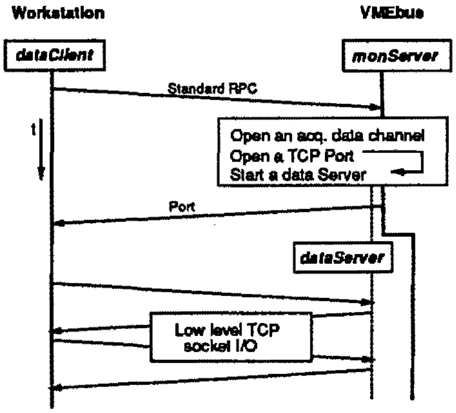


Figure 4: RPC-Less Data Tranfer

Because of their widespread acceptance, SUN 4.0 RPCs have been selected. To avoid retry problems, RPCs use the TCP/IP underlying transport protocol. The machine-dependent data format problem, unavoidable in a heterogeneous distributed environment, is solved by the *External Data Format* (XDR) layer of the RPC protocol.

The RPC mechanism is well suited to remotely control the system, but its layers introduce a time overhead that is too large to transfer a high rate of data. For this purpose, we are using a less ressource consuming protocol (Figure 4). The client contacts, with a standard RPC, the server to open a data channel on the acquisition side and a data communication port on the network side. A new specific data server, receiving both I/O descriptors, is created. The network port descriptor is returned back to the client who can establish a faster and reliable point-to-point connection (TCP/IP) to the data server to read events.

C. User Level Tasks

Users can run data processing tasks in the VMEbus system, they simply access the data through the monitoring device in the same way as for any other device. By this way the user can analyze data, build histograms, ... Data can also be copied to a disk or a tape cartridge in the VMEbus crate.

Users may want to access the data directly from processes running in their workstation. They can use a library of subroutines which takes care of the communication with the network servers in the VMEbus system. Users must provide four routines:

1. `monStart`: begins a data monitoring task;
2. `processBlock`: is executed for each block of events;
3. `monRefresh`: asynchronous user's request handling;
4. `monStop`: completes the data analysis.

Processes respond to the SIGHUP and SIGINT signals. The SIGHUP handler executes asynchronously the routine `monRefresh()` to get intermediate results while the SIGINT handler completes prematurely the data reading process.

A workstation process must indicate the VMEbus board and device it wants to read and the data channel mode. An example is the ddVME command (Figure 5), based on the well-known UNIX dd to copy data.

```
% ddVME if=vmeacq11:/dev/mon of=data.01 mode=full count=20
Warning: obs set to 1024 bytes
Connecting to 130.104.3.120(#973) ... done
Process 16478 started
ddVME: 40/0 blocks ---> 20 blocks of 1024 bytes
Rate = 24980 bytes/s
Process ddVME terminated
```

Figure 5: Example of a workstation task

Workstation Interface

An X Window interface helps the user to configure the VMEbus system and to control the acquisition processes. The interface, written in the C++ language, uses the NIH-CL and InterViews object libraries [4]. It implements Macintosh-like menus whose items are activated or deactivated according to the experiment current status. It uses dialogue boxes to get the data acquisition and monitoring parameters and to show their status. The main C++ class includes a method to dispatch user's requests and sends RPC requests to the VMEbus servers.

The user interface is able to listen to the data acquisition system and transmit its messages to the user. For that purpose, we have modified the InterViews events handler, making it sensitive to asynchronous messages from the VMEbus system.

Conclusion

We have developed a simple architecture for a data acquisition system in which real-time data acquisition, monitoring tasks and system control have been loosely coupled. This provides the flexibility of the system. The system is now fully integrated within our network of workstations and is used in experiments around the Louvain-la-Neuve Cyclotron.

References

[1] G. Bizard, G. Costa, D. Durand, Y. El Masri, G. Guillaume, F. Hanappe, B. Heusch, A. Huck, M. Mossynski, J. Peter and B. Tamain: The Belgian-French Neutron Multidetector DEMON, *Proceedings on the International Conference on New Nuclear Physics with Advanced Techniques*, Iearapetra, Crete, June 23–29, 1991

[2] The data acquisition system and its buffer system is inspired by the UNIX kernel and the buffer cache, see M.J. Bach, *The Design of the UNIX Operating System, Prentice-Hall, 1986.*

[3] Lawrence M.Ruane, Process Synchronisation in the UTS Kernel, *USENIX Computing Systems, Vol 3, 1990.*

[4] M.Sibomana, Y. Longrée, P.Mareschal, M. Nemry, A. Ninane and F.Somers, Direct Manipulation User Interfaces, in *New Computing Techniques in Physics Research, éd. du CNRS, 1990,* and references therein.

Interfacing Industrial Process Control systems to LEP/LHC

Michel Rabany
CERN
CH-1211 Geneva 23

Abstract

Modern industrial process control systems have developed to meet the needs of industry to increase the production while decreasing the costs. Although particle accelerators designers have pioneered in control systems during the seventies, it has now become possible to them to profit of industrial solutions in substitution of, or in complement with the more traditional home made ones. Adapting and integrating such industrial systems to the accelerator control area will certainly benefit to the field in terms of finance, human resources and technical facilities offered off-the-shelf by the widely experienced industrial controls community; however this cannot be done without slightly affecting the overall accelerator control architecture. The paper briefly describes the industrial controls arena and takes example on an industrial process control system recently installed at CERN to discuss in detail the related choices and issues.

I. INTRODUCTION

Computers have gained a major importance in the overall design, construction, operation, maintenance and exploitation of today's accelerators and it is not exaggerated to say that, without them, physics research would not have become what it is, and conversely that the development of computers was highly due to the needs of basic research.

Pioneering in a statistic based research domain like particle physics leads the engineers and physicists to work at the limit of what is possible in fields like electronics, mechanics, computing, materials,etc. . They have to look permanently for and to try to make profit of new promising technologies, as soon as these emerge from laboratories. They then get used to live ahead of industry in a lot of scientific fields. They finally accept as a fact of life to develop everything they need, because they do it better, tailored to their needs, with higher performance than what they can readily find.

This is the case at CERN where people have felt in the early days the potential embedded in the computers to help them solving controls problems. Many of the basic components which make up a particle factory are now well known and currently manufactured by industry. The particle accelerators are quite comparable to other industrial machines. The running of a particle factory leans mostly on domains for which industry has developed several control systems solutions. One may profit today of this opportunity, in our new era of restricted human and financial resources.

II. THE COMPONENTS OF AN ACCELERATOR

A. *Inventory of components*

Components may be classified into two categories depending whether or not they actively participate in the production of particles: the first category will be referenced as *active* in this paper and the second as *passive* (see figure 1). Magnets, RF cavities, electrostatic separators, power converters, beam instrumentation are *active* components. All play an active role in keeping an accelerator state as well as in performing the transitions from one state to another. Electricity, vacuum, cooling & ventilation, cryogenics, personnel protection, site access are *passive* components.

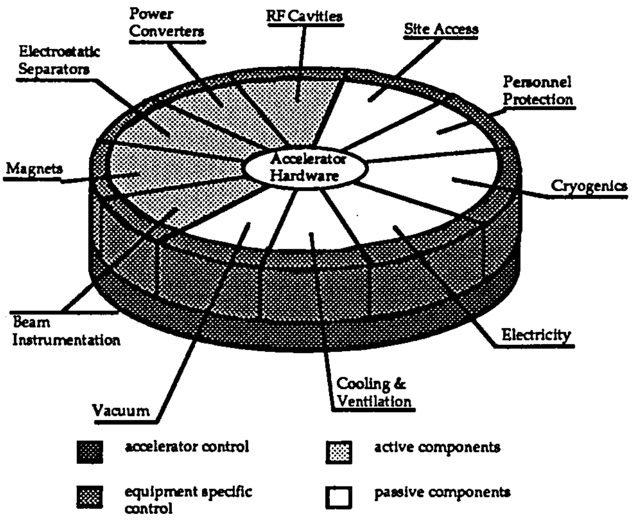


Figure 1. Accelerator components

B. The two views

Although all these components possess their own process control layer which the operator accesses through the accelerator control system, a component represents a lot of similar devices geographically distributed along the accelerator. Therefore the architecture of the overall control system has to cope with both aspects of large variety of components and wide spread of equipment along the accelerator. This explains the two views which have always developed in the past. People responsible for their equipment like to have an overview on all their equipment through component oriented consoles whereas people responsible for the control of the overall accelerator want to operate from an accelerator oriented console. There is no reason why the accelerator control system can not be designed to offer the openness which is necessary to marry these two requirements.

III. THE INDUSTRIAL CONTROL OFFERINGS

A. Domains of application

Industrial control systems have considerably developed since the seventies. They are present in industrial fields like energy production, electrical energy transmission and distribution, pharmacy, chemistry and petrochemistry, food industry, metalworking industry, paper manufacturing, glassworks, cement works, transportation, etc. . From this large variety of fields of applications, industry has gained a lot of experience. Figure 2 shows the functional layers which have been identified by industry and on which common hardware and software is built, allowing the research and developments cost to be shared between the different buyers [1].

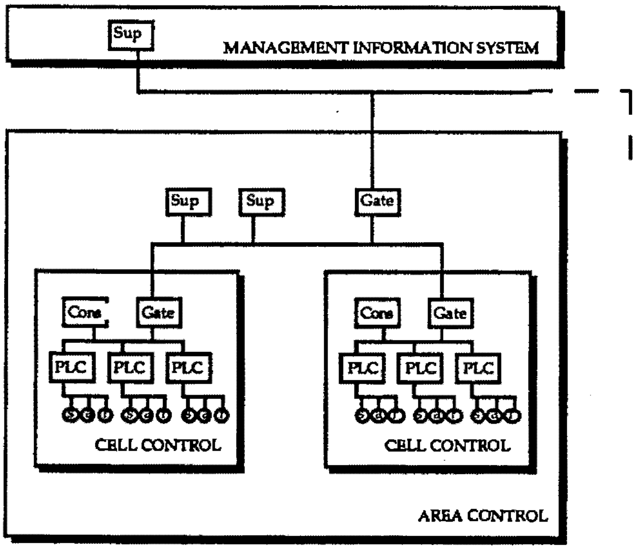


Figure 2. Industrial control functional layers

B. The different solutions

The industrial control market may be divided in two big categories.

The first one includes the Distributed Control System (DCS) supplies. They are provided by manufacturers issued from two different origins: big Programmable Logic Controller (PLC) makers or computer makers. All these suppliers offer complete control solutions with basically the same features:

- PLCs and full range of In/Out (I/O) interfaces to connect to sensors, actuators, etc.
- redundancy capability for I/O interfaces, controllers, power supplies, cabling, etc.
- environmental hardening
- local operating facilities
- multi-layered communication
- engineering tools for configuring, tuning and documenting
- color graphic operator interfaces for process control view or system overview with logging, trending, archiving and alarm management facilities
- application support including installation, commissioning, training and maintenance

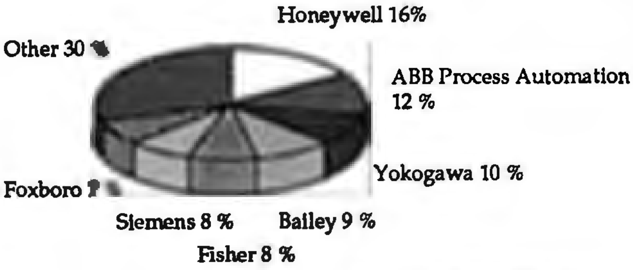
The other category of supply is aiming at simplicity and low pricing. Their manufacturers have limited their offerings to the bottom layers and have in mind the supply of laboratories, restricted test facilities, and non distributed equipment. They offer:

- PLCs with restricted capacity in number of I/O points
- simple communication (RS232, etc.)
- limited graphic operator interface for process control view, with some logging, trending and alarm facilities
- primitive engineering tools for configuring

In order to fill in the gap between this small scale solution and the complete one, other companies have developed very sophisticated process oriented application enablers software packages on different standard hardware (PCs, Vaxstations, etc.) and software (UNIX™, DOS, OS/2™, etc.) platforms They provide interfaces to most of the PLCs. They have an unbeatable openness and attempt to integrate all the possible features one might expect to help configuring, tuning and supervising. A lot of attention has been paid to the application programming facilities and environment with graphics and animation editors, all kinds of operator inputs (keyboard, mouse, trackball, touch screen), math functions, logic operations, batch functions, graphic and language base programming facilities, time scheduled events and intervals, real-time and historical trending, alarm monitoring, supervision and logging, report generation, etc. .

C. The contenders

In the first category there are less than thirty suppliers in the world and seven of them share more than two third of the market (see figure 3). The actual solutions they propose are very much proprietary. both at the hardware and software level. Due to their size and the commitment to preserve the investments of their customers, they have to keep compatibility during the evolution of their products. This is a very heavy constraint.



Source : ARC 1990 DCS Report

Figure 3. Worldwide DCS market shares

The second category is much more crowded and is occupied by hundreds of equipment manufacturers. They could care less about customers' investments as the cost of their supply is much more affordable. The market is covered by equipment manufacturers, laboratories and simple test systems. Examples of such supplier are Satt, Makmodul, SAIA, etc. . The big manufacturers of DCS are also offering products in that category. Industrial control application enablers package suppliers have names like FactoryLink™, Wizcon™, InTouch™, etc. .

D. Use of industrial controls in accelerators

Among the two categories of components in an accelerator, the ***passive*** components are the one which fit naturally with industrial control, as they run independently of the particle manufacturing process. Their processes do not fundamentally differ from those of similar equipment currently used in other factories, except for cryogenics for which there is not much experience in the world in the industrial production of very low temperature refrigerated helium.

The ***active*** components present to the control system very strong constraints. They are tightly coupled to the beam and, for an accelerator operator, are real-time process control elements. Those are normally attached to PLCs in an industrial control environment. Industry does not yet offer solutions matching large geographical spread together with tight real-time constraint.

IV. LEP ENERGY UPGRADE

A. LEP 200 new equipment inventory

LEP is the latest CERN leptons collider which came into operation in 1989. Its energy is currently limited to 50 Gev per beam but provision was made, at the design stage, for most of the equipment to allow for an increase in energy to almost 100 Gev per beam, in a second round. This energy increase project [2] is now on the way and is planned by the year 1994. The major differences with the actual machine come from the new 192 accelerating RF superconducting cavities, which will be installed in the four even interaction points of LEP . Each of this even point will be equipped with a cryoplant [3] having a cooling power of 12 kW at 4.5 K temperature. Other modifications concern:

- the replacement of the eight superconducting low-beta quadrupoles for another set of eight with a higher gradient (36 T/m→55 T/m)
- two additional electrostatic separators at each even interaction point
- new collimators at the end of the even arcs to shield the superconducting cavities from synchrotron radiation
- a few new beam position monitors
- forty new power converters, which added to the redistributed existing ones will extend the possible operation energy of the magnets from 65 to 100 Gev
- an increase in electrical power from 70 to 160 MW and its distribution
- eight new cooling towers in correspondence with the electrical power increase

B. Industrial controls for LEP 200

LEP has been completed in 1989. Although industrial controls have been sparsely used in cooling & ventilation and emerging cryogenics, their general introduction was not supported at that time by any coordinating body or task force. The problem of their integration in the overall control system was either treated as a single case or neglected. Since then, we have undertaken to study carefully the problems which will be posed by their wider introduction trying, in association with industry, to find harmonious solutions.

The first opportunity is given by the LEP energy upgrade. As may be seen from the above list of modifications, the major change concerns RF superconducting cavities and cryogenics. The other modifications are either a replacement or a minor extension of existing equipment. Cryogenics is far the best candidate for a first full industrial control implementation as it has a fair size, it belongs to the well fitted presumed category of ***passive*** components and HERA is making a similar trial with it [4]. In addition, time is just right to make this attempt as CERN is facing new economical conditions where the budget has to be kept constant while the human resources are regularly reducing towards a twenty per cent diminution level.

A specification was worked out under the responsibility of one cryogenics expert [5] who had the original idea of getting the control industry involved. It included the supply, installation and commissioning of:

- the hardware to interface to the different components of the cryoplant, i.e. compressors, cold boxes and cooled helium distribution
- the operating consoles and peripherals
- the basic system software with the various engineering tools
- the application software to run the processes and supervise the overall installation. Definition of the processes is a joined effort between the cryoplant components manufacturers and CERN cryogenics experts

The dimension of the overall cryogenics control system is in the order of 15000 I/O connections. The tender gave several positive responses, offering acceptable solutions and was adjudicated in April this year.

Although some guidelines were given in the specifications, integration in the CERN environment was left much more open as the best solution has to be found through various possibilities in a joined effort with the supplier.

The large area covered by the facilities existing at CERN has imposed the building of a large and well structured communication network on the CERN site. The basic communication system is a Time Division Multiplex (TDM) which runs over either coaxial cable or optic fibers, where the level of radiation permits it. This system follows the G703 recommendation of the CCITT [6]. The TDM offers a variety of services for the operation of the accelerator: computer networks, timing systems, data transmission, digital telephone exchange, etc..

The operation team of the particle accelerator sits in the Prévessin Control Room (PCR). The specialized cryogenics equipment will be controlled globally from a central location, the Cryogenics Control Room (CCR). The processes controlling cryogenics has to run inside computers situated in a Cryogenics Equipment Room (CER) near the equipment, all around the accelerator. Figure 4 shows the basic layout.

Supervision from the CCR needs reliable communication between the CERs and the CCR. For this purpose, the industrial control supplier was given the choice of either using point to point links via special supplier proprietary interfaces connected to private TDM channels or to make use of the computer network via the IEEE 802.5 (token passing ring) or IEEE 802.3 (Ethernet) standards. The usage of TCP/IP (Transmission Control Protocol/Internet Protocol) has been recommended. Tests of the different solutions have been planned for the end of this year in a joined effort with the supplier.

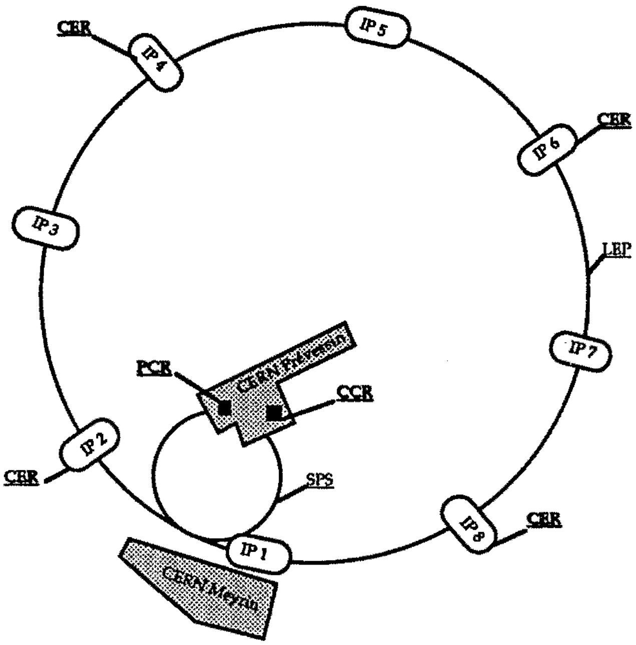


Figure 4. Cryogenics in LEP 200

Cryogenics as a *passive* component does not need very tight connection with the PCR. Cryogenics process operating states, a subset of the alarms and general diagram displays are sufficient to inform the accelerator operators of the cryogenics behavior. But provision has been made to allow for some restricted and well protected commands in order to get rid of eventual abnormal process behavior waiting to be understood and cured.

Industrial control suppliers currently offer solutions based on standard platforms to connect their system to the plant network. One μVAX 3100 running VMS and linked to Ethernet will give this possibility, by means of an access library available from the supplier, to access any equipment connected on its control system. Accelerator operation oriented data collection and restricted access will be easily implemented. All this is planned for installation before the end of this year.

V. LHC

A. LHC equipment inventory

LHC is the future CERN Large Hadron Collider to be installed above LEP in the same 27 kilometer tunnel. This machine will accelerate particles and hold them on a circular orbit by thousands of superconducting electromagnets, cooled down at a temperature of 1.8 K [7]. These will be the most challenging equipment as they require a huge cooling capacity. For this purpose, it is foreseen to increase to 18 kW the cooling capacity of the four refrigerators which are to be installed for the actual LEP energy upgrade. In addition, new refrigerators will have to be installed in the odd points. An extension will have to be added in order to lower the temperature to 1.8 K. The required 16 MV RF voltage will also be provided by superconducting cavities.

LHC is a complete new machine and requires the installation of a large number of new equipment of the *active* component category. The same requirement applies for the vacuum in the *passive* components category, but electricity, cooling & ventilation, cryogenics, personnel protection and site access will just require some extension to existing ones for LEP, if any.

The big difficulty will come from the sensitivity of the superconducting electromagnets to quenching under beam loss. Some of the *active* components will have to cooperate rapidly through computers to try to prevent this from happening [8]. It is premature to say that industrial control systems will be unable in more than 5 years from now to cope with such requirements. The question has not yet been debated with industry and has not the first priority in this preliminary industrial controls evaluation.

B. Industrial controls for LHC

The extension of cryogenics has already been foreseen in the contract with the industrial control supplier. A dimension of up to twice the size of the actual system could be envisaged without causing any problem.

Supposing hopefully that the cryogenics experience turns to be positive, the people responsible for other components in the *passive* category will have the difficult decision of opting for the extension of their process control system or for an industrial one. Important parameters of the decision will be the size of the extension, homogeneity, maintenance, training, consolidation, human and material cost, etc. . Replacement of existing LEP process control equipment to put the extension in a global industrial control solution might be even desired by the process experts but could prove to be difficult in the actual context of budgetary constraint.

Industrial control suppliers are fully aware of the importance of international standards. In the next few years, the traditional proprietary approach will be abandoned in favour of standard based solutions for all the products which will be supplied. This will range from basic hardware interfaces to operator consoles, from operating systems to software engineering applications. Openness and Management Information System will be the magic words of the next decade. The control industry and CERN have clearly a common approach which could well come out into some active collaboration in view of the LHC project.

VI. CONCLUSION

Passive components of an accelerator are the most promising equipment for which industrial controls solutions can be easily found. However if this would be rather easy to realize in a free environment, the implementation could prove much more difficult in the two major CERN projects of this decade, LEP 200 and LHC once approved, as most of the eligible new equipment will mostly appear as extensions. Nevertheless the opportunity could also come for LHC from aging equipment considerations, knowing the tremendous speed of evolution in the control domain. Meanwhile, aside accelerator prospects, there are other domains of applications in a laboratory like CERN where the industrial controls fit naturally. These are all the basic services like heat production, water distribution, general electricity (back-up electrical supply, distribution, etc.), etc. and the large bench test facilities needed to validate the numerous accelerator equipment before installation like magnets, RF cavities, etc. Basic services are already largely committed to industrial controls but their consolidation will need much more homogeneous and integrated solutions. Large test facilities have factory dimensions and are very much concerned with the traditional industrial production concepts. Industrial controls will play a very active role in all these domains in the very near future and there is no doubt that a huge experience will be gained for the benefit of our accelerators.

VII. REFERENCES

[1] Control Engineering, October 1989.
[2] C. Wyss, The LEP energy upgrade programme, Proceedings of the 1991 Particle Conference, San Francisco, USA, 1991.
[3] M. Barranco-Luque, J.-P. Dauvergne, W. Erdt, P. Frandsen, D. Güsewell, F. Haug, A. Juillerat, G. Passardi, J. Schmidt, G. Winkler, The refrigeration system for the LEP upgrade, XIIIth International Cryogenic Engineering Conference, ICEC13, Beijing, China, 1990.
[4] H.R. Barton, Jr., M. Clausen and G. Keßler, Cryogenic system for the HERA magnet measurement facility, XIth International Cryogenic Engineering Conference, Berlin-West, Germany, 1986.
[5] J.-P. Dauvergne and all, Technical Specification for the Process control system for LEP 200 cryogenics, CERN, Geneva, Switzerland, 1990.
[6] Digital networks: Transmission Systems and Multiplexing Equipment, Recommendations G700-G956 (CCITT, Geneva, 1985).
[7] Design study of the Large Hadron Collider (LHC), CERN 91-03.
[8] K.H. Kissler, F. Perriollat, M. Rabany and G. Shering, Controls for the CERN Large Hadron Collider (LHC), Proceedings of this conference.

SPS/LEP Beam Transfer Equipment Control Using Industrial Automation Components

A. Aimar, G. Bérard, J. -L. Bretin, E. Carlier,
J. H. Dieperink, M. Laffin, V. Mertens, H. Verhagen
European Organization for Nuclear Research (CERN) - SL Division,
CH - 1211 Geneva 23, Switzerland

Abstract

Several control systems for SPS and LEP beam transfer equipment have to be commissioned in the near future. Tools for fast software development, easy maintenance and modifications, compliance with industrial standards, and independence of specific suppliers are considered to be essential. A large fraction of the systems can be realized using off-the-shelf industrial automation components like industrial I/O systems, programmable logic controllers, or diskless PCs. Specific electronics built up in G-64 can be integrated. Diskless systems running UNIX and X Windows are foreseen as process controllers and local access media.

I. INTRODUCTION

Within the SPS and LEP Beam Transfer sector at CERN, there are currently several control systems being prepared for the application in both accelerators. Among those are, in different phases of progress, the control for the LEP beam dump and for the LEP Pretzel beam separators.

Whereas the size and the complexity between our different systems varies considerably - from some 10 I/O channels to nearly 1000 - all systems can commonly be characterized as 'slow controls', i.e. the required response times as seen from the main control room are only in the order of seconds for most actions. Specific fast responses, e.g. for beam dumping, are supported by special hardware. The size of data exchanged between the main control room and the equipment is relatively small. However, the significance of the equipment for the machine operation requires a continuous monitoring of its performance and efficient diagnostic tools for debugging in case of failure.

To facilitate the running of an increasing number of systems with different functionality and different composition with the available manpower we are looking for rationalization opportunities. Inspired by investigations in the field of industrial process automation several areas have been identified. In this article we try to demonstrate how we plan to apply our findings in the coming generation of control systems.

In the following we shall first discuss the criteria which we consider essential for the selection of the appropriate components. Afterwards, the characteristics of the preferred field bus will be given. Then, the full concept will be presented complemented by a brief status report.

II. SELECTION CRITERIA

A. Hardware Interface

While the prices for equipment electronics in general are decreasing, any cabling, either inside crates and racks or between racks and equipment, is becoming a dominant cost factor, besides being error prone. Therefore all data should be picked up by remote I/O systems where they arise, whenever possible and if not totally inconvenient. All I/O systems belonging to the same logical system should be hooked to a field bus connected to a process controller.

A second factor is a clean and simple rack cabling scheme including appropriate connection systems permitting a fast and cost saving installation and modifications and an easy maintenance of the electronics.

Other important criteria concern the economic use of rack space and the cost, reliability, and the technological lifetime of the selected I/O systems.

B. Local and Remote Access Facilities

Since the equipment to be controlled is, for technical reasons, sometimes far away from the process control computer or even the I/O system, one has to provide means to influence or monitor the process locally (i.e. without the need to communicate to somebody in front of the computer), e.g. for debugging or calibrating an equipment. This can be done using local hardware, e.g. panels with DVMs, switches, and potentiometers, hand-held terminals in the case of programmable logic controllers, or portable PCs. It is essential to enable the person in charge of maintenance to get access to all important parameters of the system in a user-friendly form and to permit to develop or to modify test procedures rapidly.

Adequate facilities have to be added on the process controller level and on other systems from where people have to access the equipment, which should be as complete as possible, preferentially without much adaption work for different platforms.

C. Software

The equipment software can be a particularly problematic area in that here the user requests have to be translated into specific instructions by comparatively few specialists. The consequences have intensified within the past few years with the trend towards a higher degree of distributed processing, made possible through the application of modern technology.

In fact, if reaction time is virtually no problem as in our applications, distributed processing can largely be replaced by central processing (data 'harvesting'), using sub-systems mainly as remote I/O stations. Local processing can be restricted to those tasks requiring a really tight control over the process thus reducing the communication traffic and freeing the central system, or to mask hardware peculiarities. This reduces the amount of code to be written for the communication between the sub-systems. Software development is also speeded up due to the usually superior facilities available on the central system.

The facilities for the development of the remaining sub-system code should still be adapted to the amount of code to be delivered (e.g by using cross software on PCs).

With the distances (and sometimes also the number of systems) involved the possibility to download code rather than to change EPROMs has also become mandatory.

D. Field Bus

The proper choice of a field bus is a key item in that the field bus constitutes the same for the equipment electronics as a network for the process controllers and other computers. To a certain extent it predetermines the type of equipment to be used. Generally spoken, the adherence to industrial standards widens the choice for off-the-shelf acquisition of material.

Besides the obvious criteria like reliability (recognition of transmission errors), sufficient speed and extensibility, and reasonable cost we consider a good multi-vendor support essential. The applied field bus should be able to host equally different types of industrial equipment as well as existing home-made equipment.

III. THE BITBUS

A. General Remarks

We find that the Bitbus fulfills our criteria for a field bus in a nearly ideal way and devote therefore some room for its discussion.

The Bitbus is a high speed serial control bus which was developed in the early eighties mainly by Intel [1] and which has meanwhile found a broad worldwide distribution. In fact, it has become one of the de-facto standards for industrial control applications. It is already being applied in other physics research institutions, e.g. in [2, 3, 4].

B. Basic Characteristics

Electrically the Bitbus is based on the RS-485 standard. It uses twisted pair wires for transmission with termination resistors on both ends forming a segment. Such a segment can have a maximum length of 0.3 km hosting up to 28 'nodes' when operated at a speed of 375 kbit/s. If a speed of 62.5 kbit/s is used, and with 10 repeaters, a maximum length of 13.2 km can be reached. The maximum number of slave nodes is 250. The communication is based on messages (order-reply) exchanged between nodes. The node connectivity is master-slave-slave-... with the master node usually residing in the host interface.

C. Data Integrity

A subset of the SDLC protocol (= Synchronous Data Link Control) by IBM is used as transmission protocol describing the data exchange and the access rights of the different nodes. For assuring the correctness of delivered and received data a CRC (= Cyclic Redundancy Check) method is employed.

D. Node Structure

Each node contains a 8044 microcontroller built up around the 8051 microprocessor. These controllers run the multi-tasking operating system DCX-51 contained in firmware. The communication between the master node and the slave nodes is performed at the task level, i.e. the controllers take care of the communication between nodes using Bitbus messages. In each node the task 0, called RAC (= Remote Access and Control) task mediates the accesses from the applications to the I/O port of the nodes. The DCX-51 permits to run up to 7 user tasks in addition which can be used to perform local processing if desired. This is a particularly attracting feature since one is not obliged to write any node software to read or write I/O ports, but provides a proper framework in case one needs to do something more.

E. Software Tools

The software development for the nodes can be done 'cross' on PCs under DOS (or e.g. on a workstation running a DOS emulation) either in PLM/51, in C, or in ASM/51 (assembly language). Afterwards, the code is downloaded. The allowed code size is typically about 32 kbytes.

For the communication between DOS-based hosts and the nodes exist high-level interfaces to the usual languages. Monitor programs of several suppliers allow to perform all basic functions on the nodes or to check the actual hardware configuration on the bus.

A driver to use the Bitbus on PCs running the real time operating system LynxOS as proposed for the future use in all process controllers at the CERN accelerators [5] is currently under development.

F. Multi-vendor Support

The widespread support the Bitbus has found is reflected in the number of interfaces available to different host systems: There exist links to VME, PC, Q-Bus (µVAX), Multibus-II, and the SMP-Bus, partly also from several suppliers. A SCSI interface developed recently [6] allows to connect Bitbus based systems also to workstations, for instance.

On the node level, there is also a great number of possible choices with different characteristics and form factors: I/O modules with fixed functionality, modular I/O systems consisting of a crate with a node controller and various I/O modules, programmable logic controllers (automata), or PCs used as slaves nodes. The recent development of a Bitbus node controller with G-64 interface [7] permits now to integrate also electronics based on this standard. Last but not least, piggy-back nodes are available for direct implantation into special equipment.

This multi-vendor situation is advantageous when selecting the appropriate components due to the competition between suppliers.

The recent implementation of the Bitbus protocol on the newer microcontroller 80C152 (offering enhanced possibilities, but remaining backwards compatible) indicates that this concept will retain its importance over the next years.

G. Experience

To test the feasibility of the Bitbus in our environment we installed a PC together with a Bitbus node in an equipment hall and connected them through a non-shielded twisted pair cable of 15 m length, with the Bitbus loosely traversing an area heavily perturbed by fast high current discharges from one of the SPS injection systems. The system run over several days without halting or producing transmission errors.

For test setups in the laboratory we simply use flat cable. The use under DOS is greatly simplified due to the available good software support in form of high-level language interfaces, libraries, and monitor programs.

IV. ARCHITECTURE

The full controls concept actually pursued comprises the following ingredients (please refer to figure 1).

A. General layout

On the level nearest to the equipment two major cases can be distinguished:

a) In case of dominant specific requirements (as e.g in the LEP beam dump) the I/O hardware will mainly consist out of G-64 or Eurocard form factor modules, as practised in the past for similar applications. Inter-crate wiring is to a large extent done using coaxial cables. Built-in front panel elements are used for basic local control. Diskless industrial PCs running DOS are foreseen for certain applications which require greater local processing power and better display and access facilities, e.g. fast data logging and fault detection. The use of the Bitbus as field bus is envisaged for all sub-systems needing few or no specific processing, for others the connection can be made using RS-232 links between a processor in the G-64 crate and the process controller via a Terminal server (this would have the advantage that no additional driver software is required). Necessary industrial measurement devices like digital storage oscilloscopes will be branched using a GPIB link.

b) In case of more standard requirements (e.g. for the LEP Pretzel beam separators) the I/O hardware will predominantly been built up using industrial components, preferentially in the form of modular I/O systems housed in Eurocrates and linked to the Bitbus or also to RS-232. Necessary rack cabling will be done in front-side cable channels, using ribbon-type multi-pole connectors with screw terminals. The indication elements already contained in the I/O systems will be complemented by relatively simple control panels to allow local control in case of malfunctioning of the higher levels.

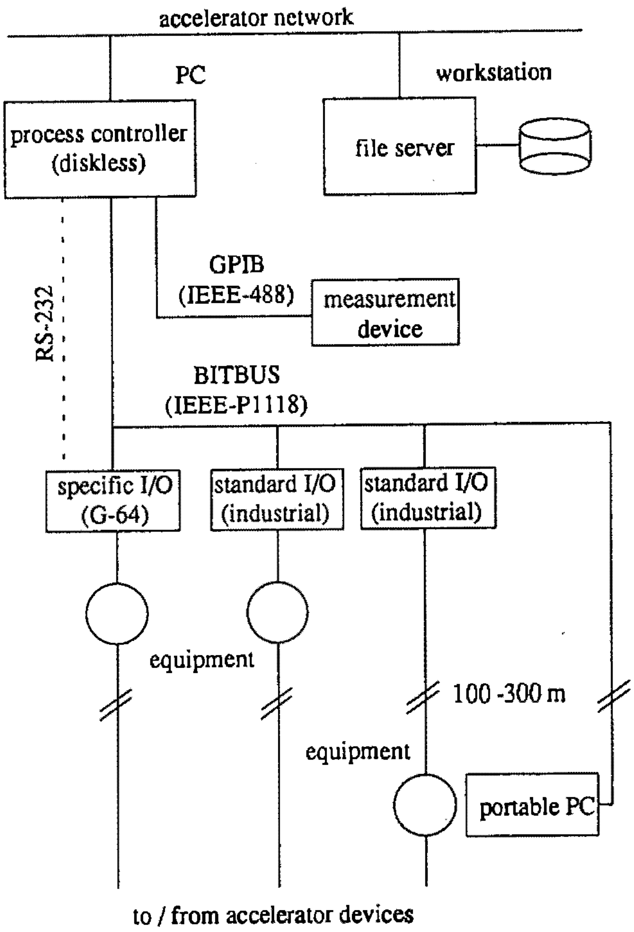


Fig. 1: Overview of the pursued concept

To provoke actions when servicing in remote areas (e.g in the accelerator tunnel) portable PCs will be used. The desired actions will be provoked by communication with the process controller via the Bitbus.

B. Node Software

The software to run on this level will be restricted to the minimum. Typical examples are routines taking care of hardware particularities or the transformation of measurements into data directly useable by the higher levels. The software for the Bitbus nodes will be cross-developed in PLM/51 on PCs running under DOS (or alternatively on workstations with a DOS emulation) and then stored on the file server for downloading. The software for diskless PCs on the Bitbus can be developed using standard PC tools like Turbo Pascal or Quick C and will stored in non-volatile memory.

C. Process Controller

All equipment specific electronics will be connected to the process controller put to our disposal by the SPS/LEP controls group as described in [5]. This device, an industrial PC running under the real time operating system LynxOS, is directly connected to the accelerator network. This PC will be diskless, possessing sufficient memory to run, besides the general software, the equipment specific control software without the need for swapping. The required code will be loaded at boot time from a file server which will also be used to store historical data for later analysis.

D. Local Control and Diagnostic Software

The implementation of X Windows on these systems lays the ground for the implementation of the required good equipment specific control and diagnostics software including graphics, to be used simultaneously for local and remote access with interaction possible through workstations, X terminals, or PCs under LynxOS or under DOS.

Such a software, at the same time aimed at grouping our various needs into one configurable package, is currently being developed and described in [8].

E. Test Setup and Status

The key ingredients of the presented architecture have been set up in the laboratory using a diskless DECstation 5000/125 to simulate the final process controller, with another workstation acting as remote file server. The Bitbus has been attached using a SCSI-Bitbus gateway. Since this device is conceived to simulate an already existing SCSI device there was no need for writing special driver software thus permitting to advance more rapidly. PCs under DOS are used in some smaller test applications and for development of node software.

V. CONCLUSION

Due to the continuous progress in automation industry, many different types of I/O systems have become available which can directly be used for slow controls applications at accelerators. Home-made equipment can easily be integrated.

The choice of the communication link to these systems is important. The use of an open, non-proprietary link like Bitbus, allows, for many of our applications, off-the-shelf acquisition of a variety of systems from different suppliers.

This permits to concentrate our efforts on system integration and on the more specific accelerator control electronics which remains to be designed in our laboratory.

VI. ACKNOWLEDGEMENTS

The continuous support of the SL/CO group and their efforts for providing the necessary general hardware and software infrastructure are gratefully acknowledged.

Many thanks go also to the DEC Joint Project Office at CERN for the generous and informal help.

The authors would like to thank the organizers of this conference for their efforts.

VII. REFERENCES

[1] INTEL, The Bitbus Interconnect Serial Control Bus Specification, Ref. 280645-001, 1988.
[2] INFN, Milano, Italy, D. Giove, private communication.
[3] EPFL, Lausanne, Switzerland, J. Lister, private communication.
[4] KVI, Groningen, The Netherlands, P. Kroon, private communication.
[5] R. Rausch and Ch. Serre, "Common Control System for the CERN PS and SL Accelerators", this conference.
[6] INCAA Computers, Apeldoorn, The Netherlands.
[7] INCAA Computers, Apeldoorn, The Netherlands, and J. H. Dieperink, CERN, private communication.
[8] A. Aimar, E. Carlier, V. Mertens, "An Open Software System Based on X Windows for Process Control and Equipment Monitoring", this conference.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S07IC03

EPICS Architecture*

L.R. Dalesio (LANL), M.R. Kraimer (ANL), A.J. Kozubal (LANL)
Mail Stop H820, Los Alamos National Laboratory Los Alamos, New Mexico, 87545
Argonne National Laboratory Argonne, Illinois, 60439

Abstract

The Experimental Physics and Industrial Control System (EPICS) provides control and data acquisition for the experimental physics community. Because the capabilities required by the experimental physics community for control were not available through industry, we began the design and implementation of EPICS. It is a distributed process control system built on a software communication bus. The functional subsystems, which provide data acquisition, supervisory control, closed loop control, archiving, and alarm management, greatly reduce the need for programming. Sequential control is provided through a sequential control language, allowing the implementer to express state diagrams easily. Data analysis of the archived data is provided through an interactive tool. The timing system provides distributed synchronization for control and time stamped data for data correlation across nodes in the network. The system is scalable from a single test station with a low channel count to a large distributed network with thousands of channels. The functions provided to the physics applications have proven helpful to the experiments while greatly reducing the time to deliver controls.

I. INTRODUCTION

EPICS is currently being co-developed by the Accelerator Technology controls group at Los Alamos National Laboratory and the Advanced Photon Source controls group at Argonne National Laboratory. Its architecture provides a wide range of functionality, rapid application development and modification, and extensibility at all levels to meet the demands of experimental physics. The hardware and software for each functional subsystem was seleceted to meet these requirements. The subsystems are: the Distributed Database, the Display Manager, the Alarm Manager, the Archiver, the Sequencer, and Channel Access [figure 1]. Technology changes, to be incorporated, will further extend the performance of the EPICS subsystems. Programs at various installations have applied EPICS successfully with no modifications to the software, demonstrating adequate performance, functionality and extensibility.

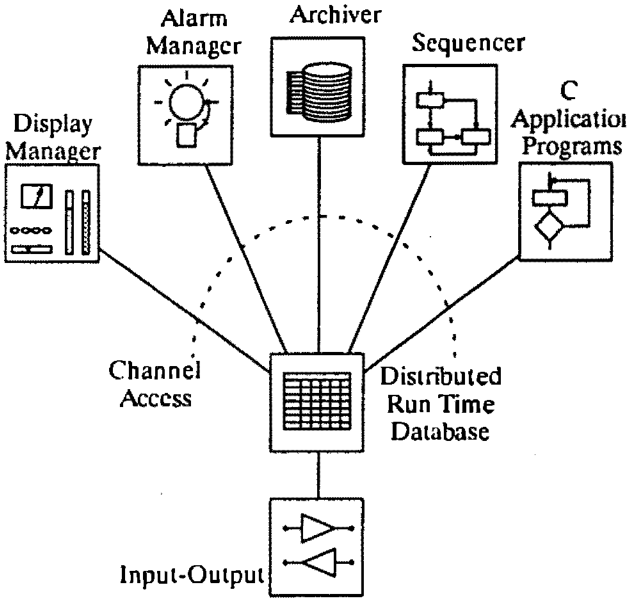


Figure 1. Functional subsystem layout of EPICS.

II. THE DISTRIBUTED DATABASE

A distributed database is used to provide local control. A portion of the distributed database is loaded in each I/O Controller (IOC). The database provides data acquisition, data conversion, alarm detection, interlocks, and closed loop control[1][2].

The IOCs, in which the database segments are loaded and executed, consist of a VME or VXI backplane, a 68020 CPU running the vxWorks real-time kernel, an ethernet connection, and an optional complement of I/O [Figure 2]. The I/O buses supported are: VME, VXI, Allen-Bradley Industrial I/O, GPIB, BITBUS, and CAMAC. The benchmarks for the database scan tasks show the ability to close 4,000 analog loops per second leaving about 40% of a 68020 available for the sequencer and channel access

---

* Work at LANL supported and funded under the Department of Defense, US Army Strategic Defense Command, under the auspices of the Department of Energy.

Work at ANL supported by U.S. Dept. of Energy, Office of Basic Energy Sciences, under Contract No W-31-109-ENG-38.

services. The maximum periodic repetition rate currently available is 60Hz. Using interrupt on end–of–conversion hardware allows for rates higher than 60Hz. The fully distributed database allows I/O controllers to be easily added to take on additional loading.

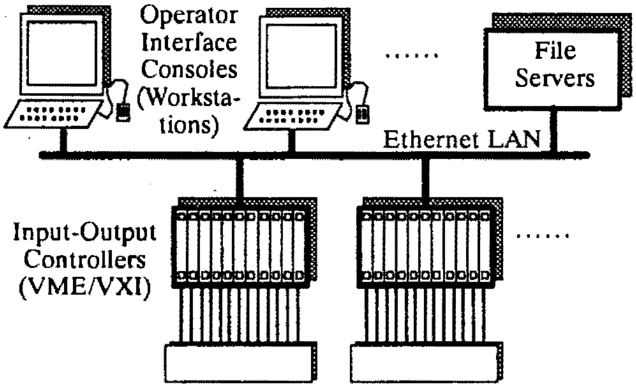


Figure 2. Physical subsystem layout of EPICS.

The database for each I/O Controller is configured offline, using the Database Configuration Tool (DCT). DCT is a menu based configuration tool written in 'C' using the Curses library. Channels are added and modified either interactively through DCT or through a text file. The commercial database package, PARADOX, has been used to produce text input to DCT. A variety of reporting facilities to help document the application is also provided in DCT. The database successfully provides data acquisition, conversion, notification, and closed–loop control without any applications programming.

The scan tasks and drivers supporting the distributed database can be easily extended. The device driver library is continually expanding to support new I/O buses and additional modules on existing I/O buses.[3] A database record is provided which invokes 'C' subroutines in the context of the scan tasks. New record types can be added to support complex device types or new algorithms. Outside the realm of the database, the IOC can be extended by writing sequence programs or tasks running directly under vxWorks that interface to the database using channel access.

III. THE DISPLAY MANAGER

The **display manager** provides an interface between the operator and the control system[4]. The display manager is capable of monitoring or modifying any field in any database distributed throughout the network via the channel access communication bus. The display manager allows the application engineer to build display hierarchies that use the windowing capabilities of modern workstations and personal computers.

The display editor and display manager run under UNIX and the X–window system. Running the display manager on the SUN IPC, display call–up of 1,000 static graphic elements and 100 dynamic graphic elements has been benchmarked at less than two seconds with an update rate of 10,000 monitor elements per second. The SPARC station I and II and graphics accelerators can be used to provide even higher performance. Using a client that is not in the same machine reduces the performance of the client by around 3%. The display manager and editor have been run on X servers on the VAX/VMS under Decwindows and the 80386AT/MSDOS under MicroSoft Windows 3.0 using the HCL–eXceed/W X–server. EPICS imposes no limitation to the number of operator interfaces that are on the control network.

The screens for display are configured using a graphic display editor (EDD). EDD can be used through text input or interactively. The X–based graphics editor provides a wide range of helpful functions for producing professional displays quickly and easily. Non–technical people have been trained to produce displays in less than 1 hour. Process displays are produced in hours. The displays are automatically connected to the operational parameters by name. No knowledge of the I/O location is needed at display configuration. Display elements are predefined for monitoring and modifying any parameter on the network, producing display hierarchies, strip charting, and plotting synchronous events. A single display can be built, then invoked many times for different instances of a similar device or subsystem. For instance, a screen can be built for one vacuum station and invoked for each vacuum station, greatly reducing the creation and maintainence of the displays. The ease of use has greatly reduced the time to create and maintain displays.

Many devices have been added to the display manager. In addition, X applications can be written to run in conjunction with the display manager. Using the X windows environment also allows users to take advantage of other packages that are available commercially and through EPICS.

IV. THE ALARM MANAGER

The **alarm manager** is used to create fault trees for alarm presentation to the operator[5]. The warning and alarms are configured in the database. The alarm manager monitors and groups these alarm conditions into a fault tree for presentation to the operator. On selecting the highest group, the alarm manager automatically traverses the fault tree until arriving at the unacknowledged alarm or the first ambiguous branch. The groups can be set up to disable on given machine parameters such as operational mode. The alarm manager provides a useful tool for steady state operation, allowing the operator to react to excursions from normal operations.

The alarm manager currently runs under UNIX. The interface to the control system is channel access. The alarm manager can be run on many workstations where each instance can run any alarm configuration.

The alarm hierarchy files are currently configured using a text editor. The format of the text files is predefined. The configuration files are converted to a binary file at initialization. The system engineer who sets up the alarm configuration can put in guidance for the operator. A screen in the display manager can also be invoked from the alarm manager to aid the operator in handling the exception.

V. THE ARCHIVER

The **archiver** is used to collect data from the control system to disk. Data collection for each data request is initiated and discontinued based on time or the condition of any parameter in the distributed database. More than one request can be active on each workstation. The archive data retriever allows users to select channels for retrieval and examination. Data that is older than one minute is available through the archiver during operation. The archived data can be plotted against time or as functions of other channels. The plots can be converted to postscript and printed, or they can be converted to encapsulated PostScript and included in word processors like Interleaf. Data can also be formatted as input to either LOTUS or EXCEL. The archiver provides a program interface into archived data as well. This feature allows the user to write data analysis codes that directly interface to the archived data. The methods for accessing archived data facilitate data analysis for experimenters and application engineers.

The archiver runs under UNIX and can collect 5,000 channels per second. The archiver uses channel access to access parameters of interest on the network. Many instances of the archiver can be run on the control network.

The archive requests are currently configured using a text editor. The format of the text file is predefined. The configuration file is converted to a binary file at initialization.

VI. THE SEQUENCER

The **sequencer** executes state programs[6][7], which run on the I/O controllers. The user can build state programs using the state notation language that are compiled and loaded into the I/O controller at initialization. The state notation language takes the I/O controller through modal changes. It can be used to switch between different operational modes (i.e. warm up, ready, operate, shut down) or to handle exceptions (i.e. vacuum leak detected – go to safe state). The state notation language has a syntax designed to make implementing state transitions easy. The state notation language features transitions for time and events. Connections to the database on the network are handled by assigning a database name to a variable. 'C' statements can be placed in the state program. Using the state notation language alleviates the need to know channel access or vxWorks internals, reducing the source code by approximately 75%.

VII. CHANNEL ACCESS

Channel access is a 'software bus' provided to communicate between various EPICS subsystems[8]. It allows the system components and user extended components to perform channel connections, gets, puts, and monitors on any field of the distributed database using the TCP and UDP protocols. Channel connections are performed by connecting a unique channel name to the IOC containing the channel. Channel access provides notification to its clients when a connection is broken and another notification when it reconnects. This connection management is used to keep all subsystems informed of the status of the other IOCs, on which it may depend for parameter information.

Channel access services are available under UNIX on the 68xxx and SPARC series processors, under VMS, and under vxWorks on the 68xxx based processors. The interface routines are in 'C'. These routines are used by the I/O controller, display manager, archiver, alarm manager, and the sequencer for communication.

VIII. EVENT SYNCHRONIZATION

EPICS is equipped with a timing system that provides event synchronization across the network. Through a hardware/software solution, events can be accurately correlated in distributed I/O controllers. These events provide data with time stamps that are used by the archiver and operator interface to accurately analyze synchronous data sets in real time and for historical data. In addition, the timing system provides the ability to process data on sub-harmonics of the base rate of the physics machine. This allows data taking to occur on the same Nth event across IOCs on the network. The timing system is designed to recover from communication failures and IOC initialization during operation. The timing system has undergone rigorous testing before being used on the Ground Test Accelerator.

IX. COMMERCIAL IMPROVEMENTS

Upgrades to the EPICS hardware and software are becoming commercially available, and require little effort to integrate. The IOC will soon upgrade from the 68020 to the 68040 processor. An FDDI backbone for the distributed network is currently available, but has not been required at any of the EPICS installations. Version 5.0 of vxWorks provides performance enhancements to the network communication, task switching, and semaphore handling. The standards upon which EPICS is built, are continually improving in performance.

X. EPICS SOFTWARE EXTENSIONS

Upgrades to the EPICS software are underway for making the system easier to configure and diagnose, and provide more functionality. Upgrades are being implemented to provide interactive X-based configuration tools for the database, archiver, and alarm manager. Extensions are planned for the display manager to include polygons. The display editor is being extended to include object grouping. Devices and database record types will be added as required.

XI. CONCLUSION

EPICS has been successfully applied to many applications[9-13]. The tools that are available have been used to provide data acquisition, supervisory control, closed-loop control, sequential control, and data archiving. All functional requirements have been met with the existing architecture. Extensions are underway to make the configuration of EPICS easier and further improve the throughput. The EPICS collaboration continues to provide useful tools for producing better results with less expense to each individual program.

XII. ACKNOWLEDGEMENTS

The initial design and implementation of EPICS was done by the Controls Group of the Los Alamos Accelerator Technology Division. In it's previous incarnations EPICS was known as the Ground Test Accelerator Control System (GTACS) and the Los Alamos Accelerator Control Systems (LAACS). EPICS is currently being co-developed by the Accelerator Technology controls group at Los Alamos National Laboratory and the Advanced Photon Source controls group at Argonne National Laboratory. Specification, design, implementation, verification, and configuration control has been performed by the following group: J.B. Anderson (ANL), M.D. Anderson (ANL), B.C. Cha (ANL), R.A. Cole (LANL), L.R. Dalesio (LANL), C.E. Eaton (LANL), B.A. Gunther (LANL), J.O. Hill (LANL), D.M. Kerstiens (LANL), M.J. Knott (ANL), A.J. Kozubal (LANL), M.R. Kraimer (ANL), F.R. Lenksus(ANL), W.P. McDowell (ANL), G. Slentz (LANL), M.E. Thuot (LANL), R.C. Zieman (ANL). EPICS continues to evolve as new perspectives are provided and new challenges are met. The evolution is directed at providing better control tools to the experimental physics community.

XIII. REFERENCES

[1] Dalesio, L. R., "The Ground Test Accelerator Database: A Generic Instrumentation Interface," in *Accelerator and Large Experimental Physics Control Systems*, D. P. Gurd and M. Crowley-Milling, Eds. (ICALEPCS, Vancouver, British Columbia, Canada, 1989), pp. 288—291.

[2] Dalesio, L. R., "The GTA Control System Database: Configuration, Runtime Operation and Access," in *Proceedings of the 1989 IEEE Particle Acelerator Conference*. (PAC, Chicago, Il., USA, 1989), pp. 1693—1694.

[3] Arnold, N.R., Daly, R.T., Kraimer, M.R., McDowell, W.P., "I/O Subnets for the APS Control System," submitted to Particle Accelerator Conference, San Francisco, California, May 6—9, 1991. .

[4] Hill, J., Kerstiens, D., and Dalesio, L.R., "A Virtual Control Panel Configuration Tool for X-Window System, " submitted to ICALEPCS, Tsukuba, Japan, Nov. 11-15, 1991.

[5] Kraimer, M.R., Cha, B.C., and Anderson, M.D., "Alarm Handler for the Advanced Photon Source Control System," submitted to Particle Accelerator Conference, San Francisco, California, May 6—9, 1991.

[6] Kozubal, A. J., L. R. Dalesio, J. O. Hill, and D. M. Kerstiens, "A State Notation Language for Automatic Control," Los Alamos National Laboratory report LA-UR-89-3564, November, 1989.

[7] Kozubal, A. J., "Automation From Pictures: Producing Real Time Code from a State Transition Diagram," submitted to ICALEPCS, Tsukuba, Japan, Nov. 11-15, 1991.

[8] Hill, J. O., "Channel Access: A Software Bus for the LAACS," in *Accelerator and Large Experimental Physics Control Systems*, D. P. Gurd and M. Crowley-Milling, Eds. (ICALEPCS, Vancouver, British Columbia, Canada, 1989), pp. 288—291.

[9] Atkins, W., "Using the EPICS Sequencer Tool to Automate the GTA Vacuum System, " submitted to Particle Accelerator Conference, San Francisco, California, May 6—9, 1991.

[10] Wilson, William L., Marcus W. May, and Andrew J. Kozubal, "Rapid Development of a Measurement and Control System for the Advanced Free-Electron Laser," submitted to Thirteenth International Free-Electron Laser Conference, Santa Fe, New Mexico, August 25—30, 1991.

[11] Knott, M.J., McDowell, W.P., Lenkszus, F.R., Kraimer, M.R., Arnold, N.R., Daly, R.T., Gunderson, G.R., Cha, B.K., and Anderson, M.D., "The Advanced Photon Source Control System," submitted to Particle Accelerator Conference, San Francisco, California, May 6—9, 1991. .

[12] Eaton, C.E., Little, C.K., Martz, V.E., Glasscock, R.B., Carlson, G.W., "GTA RF Control," Los Alamos National Laboratory report LA-UR-89-948, November, 1989.

[13] Schindler, J., Bemis, M., Gonder, J., Oberg, D., Wright, R., "RF Control and Diagnostics for the Average Power Laser Experiment (APLE)," submitted to the 13th International Free-Electron Laser Conference, Santa Fe, New Mexico, August 25-30, 1991

A Front-End System for Industrial Type Controls at the SSC

D. R. Haenni
Superconducting Super Collider Laboratory*
2550 Beckleymeade, Dallas, Texas 75237 USA

Abstract

The SSC control system is tasked with coordinating the operation of many different accelerator subsystems, a number of which use industrial type process controls. The design of a high-performance control system front end is presented which serves both as a data concentrator and a distributed process controller. In addition it provides strong support for a centralized control system architecture, allows for regional control systems, and simplifies the construction of inter-subsystem controls. An implementation of this design will be discussed which uses STD-Bus for accelerator hardware interfacing, a time domain multiplexing (TDM) communications transport system, and a modified reflective memory interface to the rest of the control system.

I. INTRODUCTION

The design of a control system for the SSC faces significant challenges arising from its immense physical size, plethora of control points, and wide range of time scales for controls. At the slow end of the time scale there are accelerator subsystems like cryogenics, vacuum, low-conductivity water, and industrial cooling water which use industrial process controls. For a proper perspective consider the vacuum controls. A recent estimate indicates that there will be 50,000 to 60,000 points divided among 200 niches around the ring. This number, which does not include vacuum controls for the injector accelerators or beam lines, is roughly comparable to the total FNAL control system. The cryogenic system is 2-3 times larger than the vacuum system. The controls for these SSC accelerator subsystems totaling more than 250,000 points dwarf most large industrial and high-energy physics control systems. The present paper describes a possible front-end system for process controls at the SSC. An earlier discussion of this system can be found in Ref. [1].

II. DESIGN REQUIREMENTS

The controls for these subsystems at the SSC must support process controls and smoothly integrate into the rest of the SSC control system. A general list of requirements appropriate for this discussion are as follows:

1. Provide industrial process controls.
2. Minimize hardware in hostile or radiation areas.
3. Use open, widely available industrial standards.
4. Prefer commercial products to custom hardware.
5. Set no limits on the number of control points.
6. Execute control functions at different system levels.
7. Support a centralized control system architecture in which the entire complex is operated from a single location.
8. Provide regional level controls to support commissioning and maintenance of large accelerator components.
9. Eliminate accelerator hardware data hiding in the control system architecture.
10. Facilitate near real time inter subsystem information sharing and implementing geographically distributed or inter subsystem control loops.
11. Contain costs.
12. Maximize reliability.
13. Follow the overall SSC control system architecture.

While meeting many of the process control requirements for individual subsystems, commercial process control systems have problems supporting either the large number of control points or extended geographical area or desired level of integration.

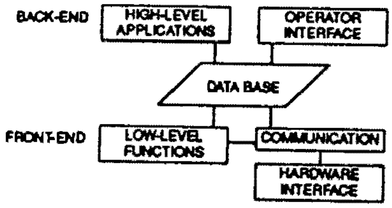


Figure 1. Front-end system components in relationship to the rest of the control system.

III. FRONT-END SYSTEM DESIGN

In Fig. 1 the basic components of the front-end system are show in relationship to the rest of the system. The front end consists of the interface hardware, communications, and processors to execute low-level control functions. It supports data I/O and functions such as loops, alarm monitoring, and interlocks. The back end is commercial and SSC developed software providing operator interface and high-level control applications such as complex sequencing, expert systems for alarm analysis, data logging, and process simulation. Effectively the high-speed reflex-like control activities are

---

*Operated by the Universities Research Association, Inc., for the U.S. Department of Energy under Contract No. DE-AC02-89ER40486.

assigned to the front end while the slower more complex processes are placed in the back end. This organization naturally allows the high-speed controls to be distributed near the I/O interface when needed. The interface between the front and back ends is shown as a database. This will likely consist of several different types of databases coupled with data access and communications management software. In operation the front end continuously updates the database at a rate which would maintain a near real-time snapshot of the control points. The back end has high-speed random read access to the data as needed by the different applications. Commands to change output settings normally must pass various sanity and permission tests before being committed to the hardware. They are best processed as messages.

A three level architecture is suggested for the front end. The global level is the central SSC control room. All data from the accelerator complex is available at this location. The global level supports a back end for all process subsystems and other higher-speed SSC controls. Low-level functions executed at this level are able to span the entire accelerator complex. The regional level would cover a part of the accelerator or a major component. A regional system would have access to the regions data and would support back ends but only for hardware covered by the region. Regionally supported operator interfaces are intended for commissioning, maintenance, and emergency backup. It is envisioned that most of the low-level control functions would be executed at the regional level. The bottom level contains the hardware interfaces. Each system can see only the I/O interfaced directly to it. The primary function of these systems is to act as a concentrator for control points. They will support minimal operator interfaces for maintenance. When needed low-level control functions can be added to an interface level system. There is 1 global level system, less than 100 regional level systems, and 2000-3000 interface level systems.

A basic feature of the SSC control system will be the use of telephone technology based communications in lieu of LAN based communications. The idea is to provide a large number of independent point to point communications paths with dedicated band width over a TDM system. There is no degradation of performance even if all paths operate at 100% capacity. The telephone industry continues to push TDM technology to higher data rates thus giving rise to the notion that communications is not a limiting factor in the design of a front-end system. For a LAN based system this is not the case. Since performance decreases with increasing load one must design a system which minimizes communications usage.

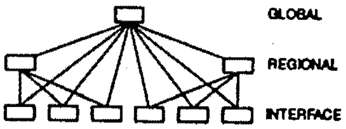


Figure 2. Front-end system communications links.

A possible set of communications links between the three front-end system levels is shown in Fig. 2. Each interface level system is connected to a regional and the global systems. Regional systems also connect to the global system. Thus the interface level delivers the same communications performance to both the regional and global levels. This would not be the case if the data had to flow through the regional system on its way to the global system. Depending on the TDM configuration this scheme allows for redundant communications paths. The role of a failed regional processor could be picked up by the global system thus providing backup processing without having to supply a duplicate of each regional level system.

The communications protocol over the TDM links may be chosen to minimize data and processor overhead. It has been decided to implement communications using reflective memory. Reflective memory is similar to a shared memory except that instead of one physical memory being shared by two processors there are two memories connected by a hardware implemented communications link. When a value is changed in one memory there is a small delay before it is reflected in the other. Hardware implemented reflective memory reduces the communications overhead on the processors to memory access. Such a link is proposed for the connection between the global and regional systems. This would allow the global system to monitor and change the parameters of the regional low-level function processing and provide a consistent basis for switching from regional to global back-up control. Note that this communications link is only for front-end operation. The regional and global systems are also connected by LANs for other types of communication.

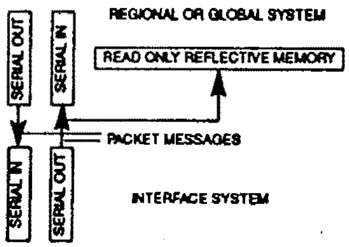


Figure 3. Communications model for the global/regional to interface level links.

Communications with the interface level systems uses a modified form of reflective memory. This is outlined in Fig. 3. Effectively one combines packet (serial) communication with read-only reflective memory at the upper level end of the link. The high-end communications interface interprets some messages as reflective memory updates and others as serial communications. The link to the interface system uses only packets. This scheme takes advantage of the above mentioned message nature of commands and the fact that control points are more often read than written. Thus a single link can serve both reflective memory and serial communications. This provides for program down loading,

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S07IC04

remote debugging, and hardware interface testing. Finally the communications interface is simplified for the front-end level which has the most systems.

The global and many of the regional systems will integrate controls from several process subsystems and/or higher-speed accelerator subsystems. Exact details of the hardware and software for these systems are beyond the scope of this paper. Of importance to the process controls front end is the need for a database and software to manage the configuration of and access to the reflective memory data. A second database would provide a repository for system run time parameters like set points or alarm limits. The former database is a constant of the hardware configuration while the latter could change dynamically during accelerator operation. These databases are extracted from a central configuration database and must be maintained consistent between all systems. This will be accomplished using communications links and software outside of the front-end system.

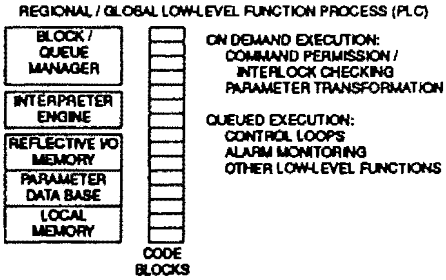


Figure 4. Components of the global/regional level server process for low-level control functions.

The global and regional systems support the front-end system process which provides a server for the low-level control functions. A schematic idea for this process is shown in Fig. 4. It would contain a large number of more or less independent code segments each of which carries out some specific control function. The code in these blocks (likely tokenized) could be executed by an interpreter which supported several specific languages for different types of blocks (e.g. ladder logic, control loops, alarm logic, data transformation, etc.). These blocks would have access to reflective memory data, parameter data, and local data which would be reflected between regional and global systems. A block manager would be able to dynamically add and delete blocks from the list, schedule their execution, and monitor block memory access. It should be possible to safely modify control strategies for various devices without shutting down the whole system to make the changes. Blocks could be scheduled for repeated execution in various queues thus forming a kind of PLC. Other blocks could be run on demand to provide interlock/permission checking or transformations from raw values to engineering units. This process should be maximized for efficient block execution by perhaps running on dedicated hardware and closely integrating with reflective memory and parameter access software to minimize execution overhead.

The hardware and software at the interface level would be entirely within the front-end system domain. It is envisioned that interface systems would be based on an open, industrial standard bus supported by a number of vendors supplying inexpensive industrial interface cards. An interface crate would contain a TDM communications card and a minimal I/O processor, IOP, whose task was to manage communications. When needed a second processor could be added which provided PLC functions and had its own communications capability. The IOP would provide data for the PLC at the same rate as the regional and global levels. Fig. 5 shows a schematic of systems with and without the optional PLC. Interface level systems would allow direct terminal attachment for bench testing and/or emergency low-level interface maintenance. Part of the general control system architecture is an operator station network that connects the global and all regional level systems with operator stations. This network will extend to all technical areas. With a portable operator console it will be possible to test the entire control system linkage to a given device while being at the device location.

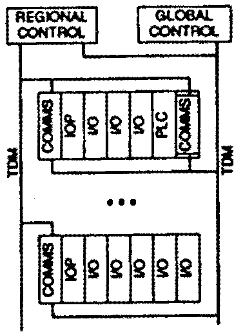


Figure 5. Schematic of interface level systems with and without an optional PLC. Connections to upper level systems are also shown.

The proposed front-end system is a combination of both hardware and software. Care must be taken in designing the software for several reasons. First there are a large number of I/O processors, regional/global level code blocks, and perhaps interface level PLC systems. Secondly, this software is very dependent on the configuration of the interface hardware. Next the system levels are linked through the layout of the reflective memory. Finally, considering the wide variation in controls at the SSC it is unlikely that there will be the uniformity between systems which would allow programs to be used in a large number of systems with only minor modifications.

For these reasons it is impractical to consider programming each system separately in traditional ways. The software generation must be automated to a large extent. One way to accomplish this is shown in Fig. 6. Here the main technical database which holds the configuration of the accelerator and control system forms a data source to automate the program generation. This database provides the configuration of the reflective memory to the regional/global systems and provides crate configuration, I/O attachment, and reflective memory update message format information to an intermediate compiler which generates the IOP code. Code blocks and PLC programs could also be captured in the database through macro statements again with the code being regenerated through appropriate translators. As suggested in the figure, users would not directly modify programs but rather change the database which in turn modifies the front-end software. Such a scheme can effectively enhance code reuse, simplify programming, and provide a single data source for describing the system. Some of these ideas have been successfully used in an earlier cyclotron control system[2].

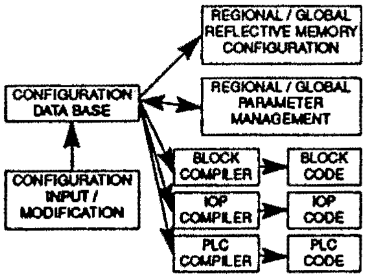


Figure 6. Possible method of automating front-end system software development through the use of a configuration data base

IV. TEST IMPLEMENTATION

A prototype of an interface level of this system is under development at the SSC. The initial system will be based on STD-Bus, IEEE-961, which is an inexpensive, open, industrial standard bus that has been used extensively for control applications in many areas. It is a well established bus supported by a large number of vendors. While initially an 8 bit bus, recent extensions to 16 and now 32 bit protocols give indications that the bus will not quickly fade or be replaced in the time scale of the SSC control system. The bus specifies a straight forward interface to the backplane thus greatly simplifying the construction of custom interface cards when needed. The bus normally uses processor cards having INTEL 80xxx or equivalent cpu chips. This gives direct access to the large amount of excellent and inexpensive development software available for the PC platform.

A dual channel STD-Bus communications card which can extract a 64 kbit/sec data stream for a T1 communications line has been designed, constructed, and is undergoing tests. This will allow up to 24 STD-Busses to be connected to a single T1 line.

An estimate of the ability of an STD-Bus system to provide communications has been made. Assuming a 64 kbit/sec communications band width and a crate with 128 analog input signals attached to standard 32 channel multiplexed 12 bit ADC cards, an 8 Mhz 8088 processor could update 256 bytes of reflective memory in both a regional and global system at 20 Hz using only 34% of the cpu. The output link is saturated to about 70%. This example was chosen to show worst case throughput since the cpu must manage the ADC multiplexing. Quantity one pricing for this system consisting of 4 ADC cards, cpu card, communications card, crate, and power supply is on the order of $4000.

Several STD-bus manufacturers produce slave processors which have shared memory on the STD-Bus but cannot initiate backplane data transfers. These systems have ISBX I/O connectors and the communications board can be easily modified to become an ISBX daughter board. This hardware configuration could support the local PLC shown in Fig. 4. PLC software which is portable to any INTEL 80xxx embedded system is available commercially[3].

The initial regional/global system TDM interface will be a commercial communications processor which directly interfaces with a T1 communications line. Enhanced communications interface cards capable of connecting to higher rate TDM lines will be developed as part of the general SSC controls system.

Initially the I/O processor will be directly programmed to produce an embedded system without an operating system or real time kernel. One option is to embed a FORTH kernel in the processor and program it over the communications line. Other options include more traditional embedded code development using assembler, C, or C++.

V. CONCLUSION

A scheme for providing highly integrated process controls for the SSC has been described. It makes use of a different style of communications based on telephone technology, supports a high-performance centralized architecture while allowing regional systems to exist for commissioning and maintenance. System prototypes are being implemented.

VI. REFERENCES

[1] D.R. Haenni *et al.*,"The SSC Field Bus: A High-Performance System Front End for "Slow" Accelerator Controls", SSCL-391, 1991.

[2] D.R.Haenni *et al.*,"Personal Computer Control System for the TAMU K500 Cyclotron", in B. Martin and K. Ziegler eds., *Twelfth International Conference in Cyclotrons and Their Applications*, World Scientific, 1991, pp 264-267.

[3] CPI Industrial Software, P.O. Box 1046, Bettendorf, Ia. 52722, USA.

The Influence of Industrial Applications on a Control System Toolbox

Peter Clout
Vista Control Systems, Inc.
134B Eastgate Drive, Los Alamos, NM 87544, USA

Abstract

Vsystem is as an open, advanced software application toolbox for rapidly creating fast, efficient and cost–effective control and data–acquisition systems. Vsystem's modular architecture is designed for single computers, networked computers and workstations running under VAX/VMS or VAX/ELN. At the heart of Vsystem lies Vaccess, a user extendible real–time database and library of access routines. The application database provides the link to the hardware of the application and can be organized as one database or separate databases installed in different computers on the network. Vsystem has found application in charged–particle accelerator control, tokamak control, and industrial research, as well as its more recent industrial applications. This paper describes the broad features of Vsystem and the influence that recent industrial applications have had on the software.

I. INTRODUCTION

Developing control systems for physics research from basic components requires considerable effort and the acceptance of risk. Now, software products that can form the basis of an experimental physics control system are entering the product market. This paper describes one such product. Software products exist that were developed originally for factory automation but experience has shown that these products do not network well and are difficult to apply in a research environment, often costing more in total than a home–written system! Vista Control Systems has gone the other way—finding industrial markets for control system software originally developed for physics research. These new industrial applications have added to the flexibility of the Vsystem software in a way that benefits all users without threatening the flexibility and openness of Vsystem.

Vsystem consists of several modules: Vaccess, Vdraw, Vscript, Valarm, Vlogger, and a number of utilities.

II. VSYSTEM'S REAL–TIME, NETWORKED DATABASE

The architecture of Vsystem requires that all components of the application, be they supplied with Vsystem or user–written, communicate only through the Vaccess database. A system's Vaccess database is defined by ASCII files and installed as global sections. The overall application database is usually made up from individual components distributed among the computers of the system. Collectively, the databases are a data model of the application, modelling both the actual connections and derived data.

The Vaccess database has many real–time features including change notification, alarm and warning checking, dynamic linking of hardware access and data conversion subroutines, and automatic I/O. There are many fields in each channel, or record, of the database that allow the modelling to be complete. Each channel is known by a name up to 40 characters long which is defined by the project. Database channels support most data types as either single valued or array channels. Table 1 lists the channel fields of Vaccess V2.2.

Channel Fields	Channel Name Current Value Current Value Significant Change Equipment Limits Clipping Enable Channel Data Type Read Only Enable, Constant Channel Channel Text Label Interest Count
Conversion Fields	Conversion Subroutine Name Conversion Enable Conversion Parameters (10) Built–in Linear Conversion Parameters (2)
Alarm Fields	Alarm and Warning Checking Enable Alarm and Warning Limits Alarm Label Alarm Type (Integer Channel)
Hardware Fields	Automatic I/O Enable I/O Type Hardware Type Hardware I/O Subroutine Name Hardware I/O Function Hardware Parameters (10) Survey Rates Automatic Survey Enable Input or Output
Display Fields	Display Limits Text Display Format Data Units

Table 1. Vsystem Database Fields

A. *Database Access Routines*

A library of access routines allows full access to the run–time database. Routines are included to search the database in various ways and to request and cancel change notification by wake–up or interrupt routine (AST) execution with a parameter. Change notification can be requested on the change of any field in the database. The library of access routines handles the network in a manner transparent to the user. Network access can be synchronous, leading to simpler but slower code, or asynchronous, leading to code a little more complex but up to 15 times faster [1].

In physics data acquisition and control systems redundancy is quite rare and problems are fixed as necessary. In some industrial systems, spare sensors and sensor channels are maintained and when wiring changes are made these changes must be reflected in the software at the closest level possible to the hardware change. Access routines incorporating redundancy issues have been added to Vaccess for future release to all customers.

III. CREATING GRAPHIC DISPLAYS

It is fair to say that there is a broader range of people who will use industrial systems and that the operator interface has to reflect this. In industry there is often a strict categorization of jobs with the result that a system that requires the use of a keyboard to operate it is unacceptable. These considerations also added the requirement for run–time support for trackballs and the options necessary to restrict or completely eliminate the possibility of modification of the system at run–time. For another application, flashing colors were added as an option to the Vdraw color palette.

Another industrial testing project required complete, networked program control of the Vdraw windows displayed on a workstation and control of several of the tools in each window. Thus, the Vscript process (Vsystem scripting facility) which was sequencing the test could also determine the windows being displayed to an operator. Therefore, in this case the workstation or X–terminal has no requirement for a keyboard or mouse.

B. Complete Graphic Toolbox

Vdraw, under Xwindows, contains a complete set of drawing, editing and windowing tools. Users can create both prototype control screens and control screens ready for application. Vdraw features an intuitive, easy–to–use graphics toolbox, Figure 1, with which users can create data acquisition and control windows rapidly, without programming.

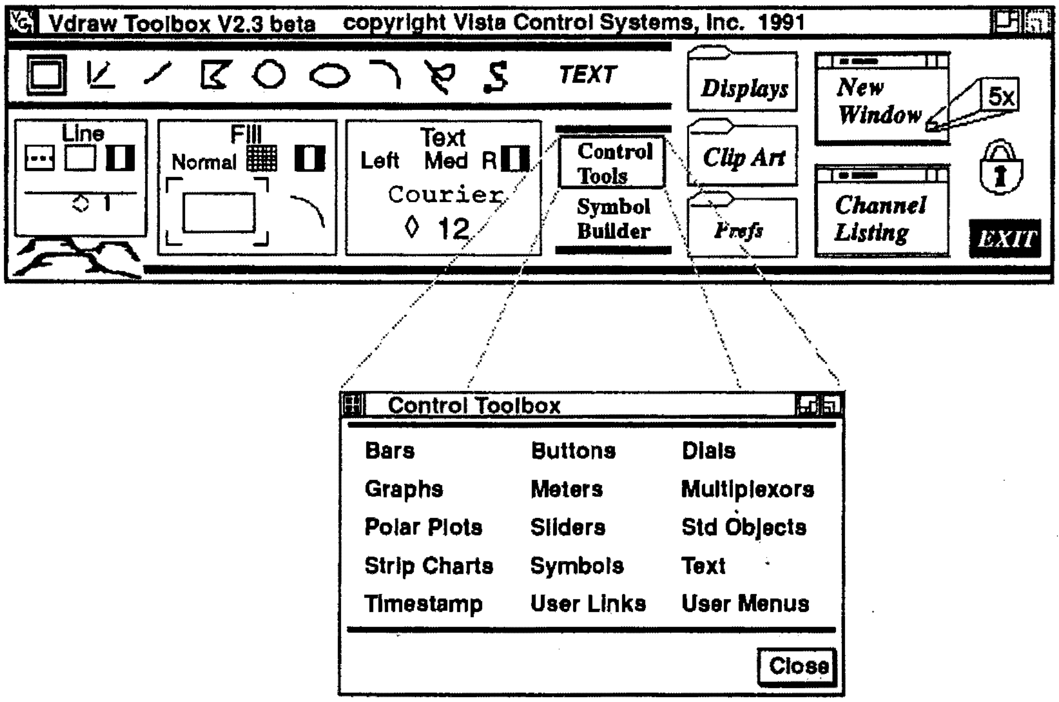


Figure 1. Vdraw's Graphics Toolbox and Control Tools Menu

In draw mode, both passive graphics and interactive graphics (known in Vsystem as *control tools)* are placed in the window. Control tools can display data from the Vaccess database, allow for changes of data in the Vaccess database, or both. The exact form of the control tool and the database connection(s) are defined with a Vdraw generated form. Vdraw also has full window editing capabilities.

When a window is placed in active mode, Vdraw connects the control objects to the Vaccess database and they immediately show the current state of the Vaccess database and hence the application.

C. Flexibility

Because in industrial systems the same window could be used in different applications—except for the actual configuration of some of the tools, such as menus—the definition of a menu can now be defined by a file rather than be hard-coded in the window.

D. Rapid Prototyping

Many industrial systems are implemented by a vendor for the client. With a toolbox like Vsystem, it is possible to develop a realistic demonstration as part of the sales effort. This has been done on several occasions with good success. An example done in-house is shown in Figure 2 on the following page.

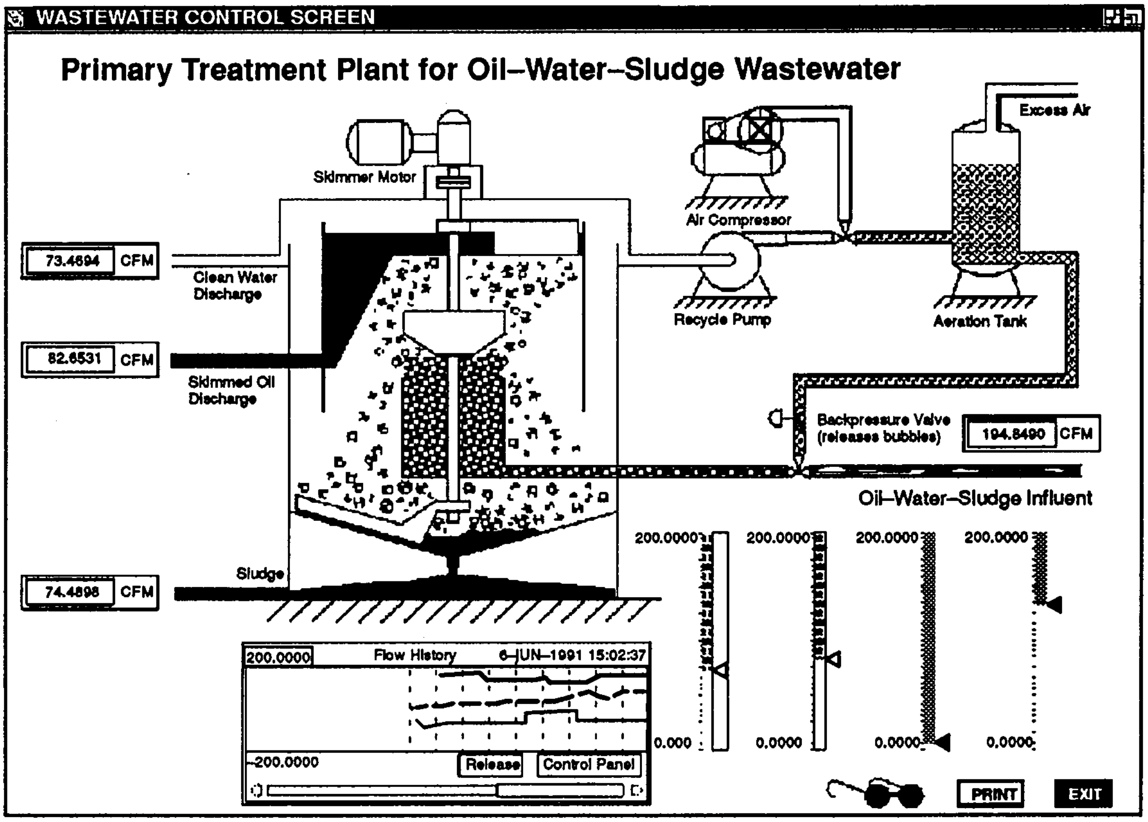


Figure 2. This control screen was created in one day by an experienced Vdraw user. It illustrates the *text*, *bar*, and *strip chart* control tools used to monitor and control the flow of the process.

IV. CONNECTING TO THE HARDWARE

Connecting the Vaccess database to the CAMAC or any other hardware is as simple as including the name of a handler when defining the Vaccess database and supplying the hardware address (branch, crate, station and sub-address in the case of CAMAC). An example handler is provided in source form with Vsystem. From this template, handlers are easy to develop.

Vscan, a generic reader for input channels, is also supplied in source form so that it can be used as a basis for application specific needs. One of the advantages of the change notification feature of Vaccess is that the real-time database can maintain an interest count and then readers like Vscan can change the scan rate of a channel depending on the level of interest in the channel.

V. LOGGING AND PLAYING BACK DATA

Vlogger, Vsystem's data logging component, reads data from the Vaccess database and writes the data to standard output devices such as disk files. Vlogger will also play data back into the Vaccess database. This playback function can include all the channels logged in a log file or just a selection of channels. Data can be logged at specified time intervals, on changes in channel value, or by an event defined by a binary channel. Logging can be gated by a binary channel. The list of channels being logged by Vlogger can be increased or reduced dynamically.

Vlogger includes facilities to take a "snapshot" of all or part of the channel settings in a Vsystem database and restoring those settings at a later time.

Vlogger is based on the client–server model and is fully networked so that a user can connect to any logging process in the system without logging on to that system and obtain the status of that logging task or modify the logging in any way.

VI. MONITORING ALARM/WARNING STATES AND ERRORS

Valarm, the interactive alarm module of Vsystem, monitors channel alarms and exceptions. Valarm will monitor all channels in an application or a defined subset. Users acknowledge and reset alarms simply by clicking with the mouse.

Alarm data can also be logged to a printer, to a disk file, or, as the result of a recent industrial requirement, to a DECtalk unit.

VII. USING SCRIPTING FACILITIES DESIGNED FOR NON–PROGRAMMERS

The first industrial application required that there be a simple way to define the sequences of operations required to collect data from an experiment or to control a process. Vscript, as described above, was the result and it is being heavily used by two industrial customers.

Vscript provides a programming environment in which non–programmers can use scripting facilities to define sequences of operations. Users interact with the Vaccess database using a simple English–like syntax that allows anyone to develop scripts. The Vscript syntax is keyword based. For example:

```
Raise TANK_12:PRESSURE to 1100 psi

                 or

TANK_12:PRESSURE should be 1100
```

Both these lines are acceptable and would put a value of 1100 into the channel named TANK_12:PRESSURE.

Vscript supports a full range of mathematical functions, file I/O capabilities, control flow constructs and logical operators. Vscript also contains several error handling facilities and supports an interface to the operating system command processor for running other programs or processes.

VIII. INDUSTRIAL INFLUENCES

E. Performance

One would imagine that it would be the research environment that would emphasis performance but in our experience so far it has been in the industrial testing environment where overall performance has been emphasized over function. This has involved the re–writing of many key Vaccess subroutines for performance and adding new routines that further reduced the per–update overhead. We have achieved performance increases of up to six–fold by this effort.

Vdraw display update rates have also been dramatically increased for some of the key tools. Here, similar performance increases have been achieved by careful use of X–windows and in one case offering the user a choice of X–window display techniques.

F. Reliability

The tolerance to software and other errors in industrial applications is much lower than in the research environment. A consequence is that industrial applications are more fully tested and problems are reported for correction. This has resulted in a continuing decrease in the number of errors in the released software and an improvement in the error handling.

G. Report Generation

The reporting from an industrial system is vital to the corporation. This might be the paper report of the shift or test or the reporting to the company database as part of the manufacturing system. To support this we have added mailbox options to some of the tools and we will be extending the capabilities here over the coming months to provide standard interfaces to common database/report systems.

H. Port to a Real–time Kernel, VAX/ELN

While VAX/ELN is almost unknown in experimental physics it has many enthusiastic users in the industrial market. This is because it is extremely well integrated with the development environment on VAX/VMS and well supported by Digital Equipment Corporation. More recently, the availability of a real_time VAX (rt300) and a component to mount on a controller board has resulted in the rt300 VAX with a VAX/ELN license becoming available in VME and CAMAC as well as in other I/O systems. It was these developments that encouraged the porting of Vaccess (and soon the other components of Vsystem) to VAX/ELN.

IX. SUMMARY

With the industrial applications, Vsystem has grown in capability, flexibility, and consistency in ways that benefits all applications, including physics applications. All of the areas of influence described above are of value to the physics community and most are of considerable value.

The strength of Vsystem lies in the X–windows graphics, the networking, the openness and the broad applicability of the system. The influence of the industrial applications has added considerably to the system. For the future we are developing new automation tools and new ports of the package.

X. REFERENCES

[1] Network Performance for Graphical Control Systems, Clout, P., Geib, M., Westervelt, R. *These Proceedings*

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC01

ARCNET as a Field Bus in the Fermilab Linac Control System

M.F. Shea, R.W. Goodwin, M.J. Kucera and S. Shtirbu
Fermilab†, P.O. Box 500
Batavia, Illinois 60510

Abstract

Data acquisition hardware in accelerator control systems is connected by a field bus to networked computers that supply data to consoles. Industry attempts to standardize on a low level field bus have not succeeded in providing a single well-supported bus. This paper describes a data acquisition chassis that connects to VMEbus computers using ARCNET, a full featured token-passing local area network, as the field bus. The performance of this technique as implemented in the control system for the Fermilab Linac is given.

I. INTRODUCTION

A field bus is the connection between local computers and the interface hardware that controls accelerator equipment. Various field buses have been used by control system designers, but no single bus has emerged as a widely used standard. Example field buses are MIL-1553, Bitbus, and CAMAC.

This paper describes the use of ARCNET as a field bus in the Fermilab Linac Upgrade project. The output energy of the Linac injector is being increased from 200 MeV to 400 MeV by replacing the last four of the nine 200 MHz Alvarez tanks with 800 MHz side-coupled structures. This upgrade doubles the beam energy in the same accelerator enclosure. The control system being designed as part of the upgrade, will be installed for the preaccelerators and the remaining five 200 MHz tanks as well as the new 800 MHz sections.

II. ARCNET AS A FIELD BUS

If the interface hardware that connects to accelerator equipment is dumb, then the field bus must be register-based. Such buses have limited addressing capacity and the amount of data transferred per transmission is small. Nodes on register-based field buses usually operate as slaves that must be polled by a master computer to collect their data.

If the interface hardware contains a processor, then a local area network can be supported. Benefits of using a local area network include longer messages, higher data transmission rates, longer distance connections, and peer protocol communications that allow a remote node to source a message without being polled. A single message may contain all the analog and digital data collected by a node.

Characteristics of ARCNET include:
- 2.5 MHz serial bit rate
- 508 byte maximum message length
- Bus, multiple star and daisy-chain topology
- Coax, twisted pair and fiber optic transmission media
- Maximum distance without repeaters >600 m. (coax)
- Transformer isolated cable drivers
- Deterministic token passing access protocol
- Built-in acknowledgment of successful message reception
- Network configuration and timeouts implemented in controller chips

Compared with the characteristics of ARCNET, typical field buses are much lower in performance: MIL-1553 operates at one MHz, has a maximum message length of 32 words and can address 32 locations per node; Bitbus is not transformer coupled and is limited to 375 kHz for reasonable distances; normal CAMAC commands access only three bytes. All of these buses use master-slave communication protocols.

III. ARCHITECTURE OF THE LINAC SYSTEM

The overall design of the Linac control system follows the standard architecture of local VMEbus 68020 computers networked to each other and to console workstations by an IEEE-802.5 Token Ring. A separate ring is used for the Linac VMEbus stations and the connection to the central control system is made by a commercial multiport Token Ring-to-Token Ring bridge. This organization isolates the traffic on the Linac ring from all other network traffic.

A. System Hardware

Individual VMEbus computers, called Local Stations, contain a core set of five cards; a 68020 processor, a 1MByte non-volatile RAM card, the Token Ring adapter, a crate utility card and an ARCNET adapter. All but the crate utility card are commercially available. Each VMEbus station acts as a stand-alone control system complete with its own database and software resident in the non-volatile RAM. Following a power outage, these systems reset, reconnect to the network, and send the last known settings to the hardware. Because they have non-volatile RAM, Local Stations do not require routine downloading. Local Stations run synchronously with the 15

---

† Work supported by the U.S. Department of Energy under contract No. DE-AC02-76CHO3000

Hz cycle rate of the Linac. Each cycle they read data from the hardware according to entries stored in a data access table, update a data pool of current readings, respond to requests for data by returning answers over the network, scan the newly acquired data for alarm conditions and send alarm messages to a multicast address on the network.

As with most control systems, the least standard part is the interface to the accelerator components. Accelerator equipment is usually sparsely scattered over a large area. The cabling length should be minimized for reasons of signal quality and cost, but the signal density is not high enough to justify the installation of VMEbus crates at all the locations where data is to be acquired. The solution is to collect equipment signals locally with cost effective interface hardware and to transmit this data to the VMEbus crates over a field bus. For the Linac Upgrade, a data acquisition chassis called the Smart Rack Monitor (SRM) collects data from the hardware and transmits it to the VMEbus Local Stations over the ARCNET field bus.

B. System Software

Network software was designed to support the Token Ring chip set. Network frames are received into a large circular buffer via DMA, and a pointer to each message contained within each frame is passed to the designated task using a standard message queue supported by the pSOS operating system kernal. Network messages to be transmitted are passed by reference through another message queue to the network transmitting logic. When the queue is flushed, consecutive messages destined for the same node are combined into a common frame if possible. This can greatly reduce network software overhead for both sender and receiver.

ARCNET support was added to the system software retaining the overall plan described above. Separate I/O buffers are used for each network. The ARCNET interrupt routine transfers the frame between the hardware buffer and system memory analogous to the Token Ring's DMA transfer. Emulation of the Token Ring software data structures permitted much of the network code to be common. A range of 16-bit node numbers is mapped onto ARCNET nodes so that user code sees no difference between the two networks. The task-task protocol widely used in Fermilab accelerator control systems allows for future protocol additions. A simple data acquisition request protocol was designed for use with the SRMs.

To support SRM data acquisition, three new entry types were designed for use in data access table entries. The first one results in sending a cycle request message, normally broadcast, to the SRMs. Each SRM that receives this message reads its own hardware data according to entries in its own data access table and replies with a single frame that includes all types of data. The second entry type waits for a given SRM to reply, and includes a deadline time within the 15 Hz cycle. One can optimize data collection by ordering the wait-type entries so that lightly-loaded SRMs appear early in the table. The third entry type is used to map the various data types included in the SRM's response into the local station's data pool. Status bits are set or cleared according to whether a given SRM replied in that cycle. These status bits can be enabled for alarm reporting if an SRM stops sending replies.

IV. SMART RACK MONITOR

The SRM is an extension of the MIL-1553 Rack Monitor (RM) used in the DZero colliding beams detector. In that case, the RM was dumb, so the register-based field bus protocol MIL-1553 was used. For the Linac Upgrade, it was decided to use the existing digitizers and D-A converters. Operating these devices requires that a computer be located close to the equipment thus requiring a processor in the rack monitor.

A. SRM Hardware

The SRM, shown in Figure 1, is a 2U chassis that contains 64 16-bit differential A-D channels, 16 12-bit D-A channels, 8 bytes of digital I/O, 256K bytes of nonvolatile RAM, the ARCNET interface and a processor. The processor is on a Motorola *Business Card Computer* (BCC) that includes an MC68332 microcomputer, 64k bytes of RAM, 128k bytes of ROM, and a serial port. The BCC is marketed by Motorola as part of an MC68332 evaluation kit. Three BCC sockets were included on the SRM motherboard: two for expansion daughterboards and one for the BCC. Daughterboards used in the Linac system will be for special equipment interfaces and for the timing system that is based on a 10 MHz serial event clock. A BCC and a 9-byte digital I/O daughterboard are shown in Figure 2.

ARCNET was chosen for the SRM network for several reasons: its cost is low, its 2.5 MHz serial bit rate is adequate, the physical bus/logical ring architecture makes it easy to connect and the token passing protocol eliminates collisions. The automatic network configuration minimizes processor overhead, the *free buffer enquiry* and *message received* features of the protocol make the communication reliable. Silicon support for ARCNET is excellent.

B. SRM Software

At this time, the software in the SRM is limited to data acquisition and debugging, although future applications may include additional tasks such as closed loop regulation. Software in the SRM is written in C and uses the pSOS real time operating system kernel. Motorola's 332Bug program that is bundled with the BCC is retained for low level debugging.

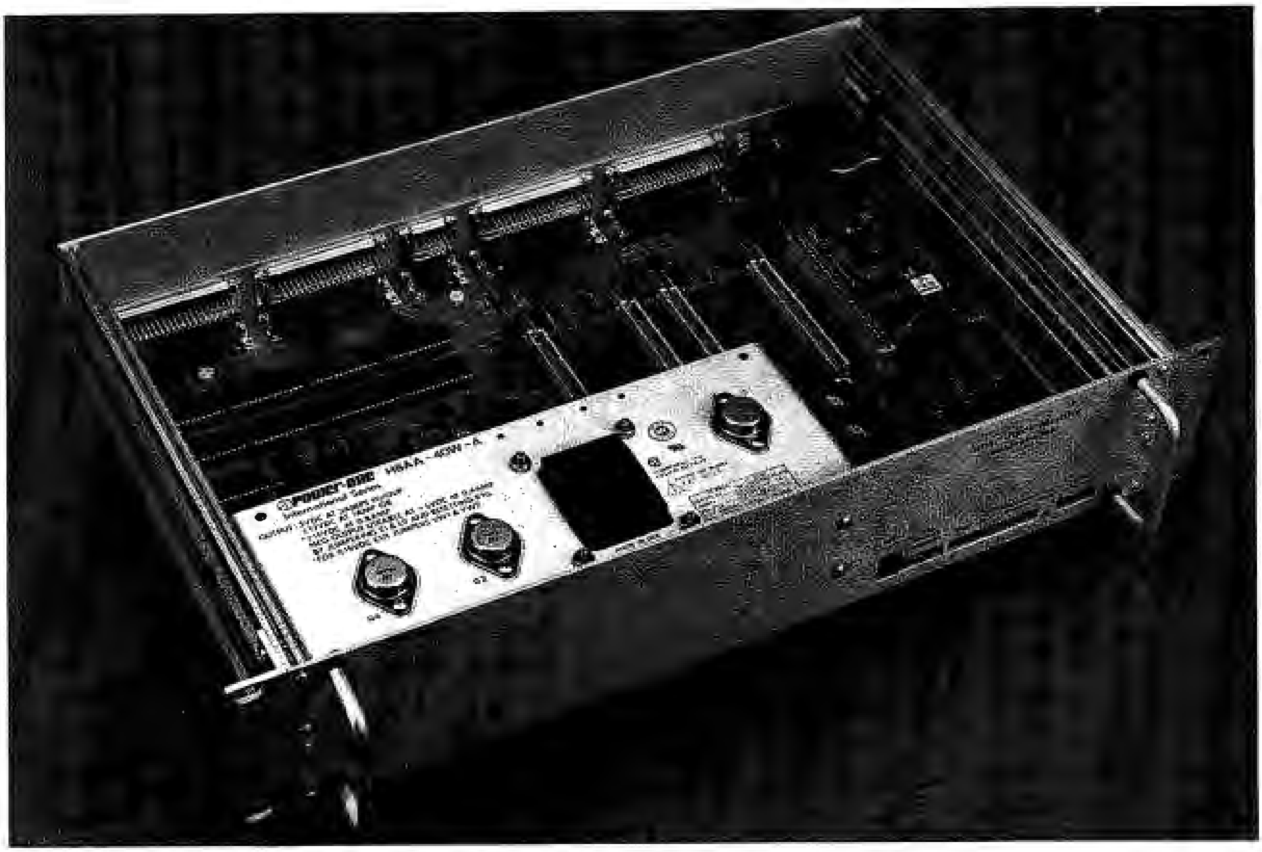


Figure 1. Photograph of a Smart Rack Monitor

When the SRM is reset, 332Bug initializes the MC68332 chip selects and then enters the SRM system code where it executes a repetitive cycle that collects data and prepares a message to send to the VMEbus Local Station over ARCNET. Normally, the cycle is initiated by the ARCNET broadcast message sent by the Local Station, but an internal M68332 timer is used to trigger a cycle, if no ARCNET message is received, to permit stand alone testing.

Figure 2. Photograph of a Business Card Computer and a 9-byte Digital I/O Daughterboard

A copy of the system code resides in the 128 kbyte PROM on the BCC, and another copy may be stored in the non-volatile RAM of the SRM motherboard. During initialization, if a RAM version is found, that copy is used. If no system is stored in RAM, the PROM version is copied to RAM. The MC68332 contains a hardware watchdog timer that requires refreshing at regular intervals. If a timeout occurs, the PROM version of the SRM code is copied to RAM and operation resumes. New versions of SRM code can be downloaded over ARCNET. Because some version of the SRM program is available in PROM, the SRM will function at least well enough to allow downloading the current version of the software. In normal operation, the SRM will find the current version in RAM.

The SRM determines what data to collect by referring to a data access table containing 16-byte entries that describe the type and amount of data to read. Separate data types are defined for digital I/O, A-Ds, D-A readback and so on. After all data is read, the SRM replies to the Local Station's broadcast request. Note that all SRMs connected to a Local Station, usually three or four, acquire their data in parallel and the Local Station only receives one message from each SRM.

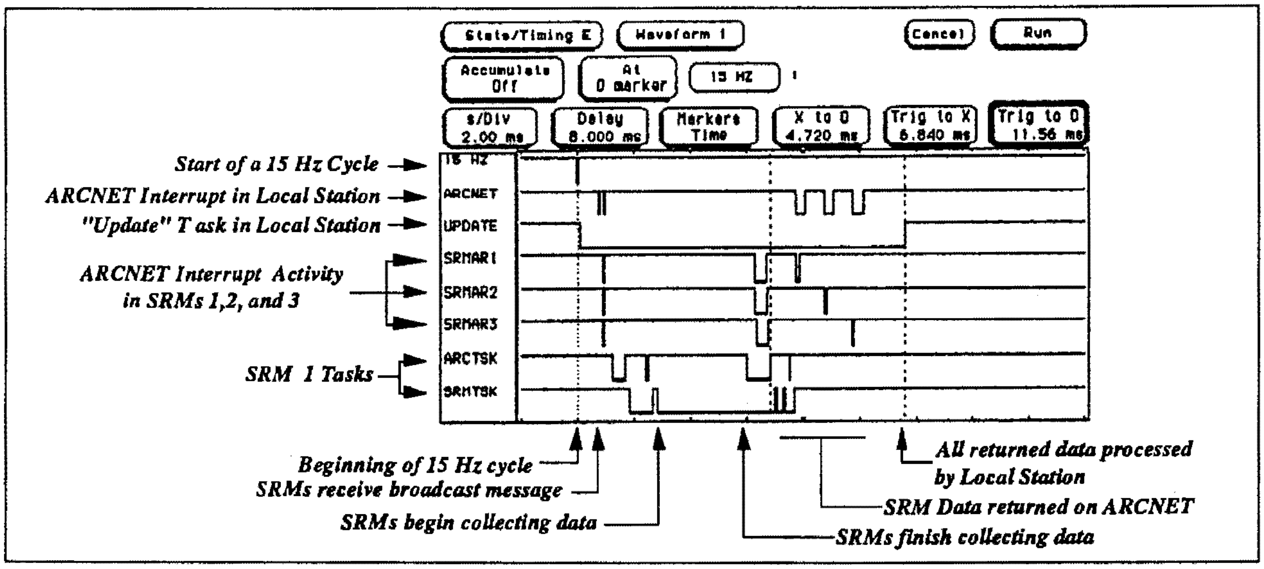


Figure 3. Logic State Analyzer Timing Display of SRM Operation

Data types are included to read back setting values of digital and analog output in the message returned to the Local Station. This is done to allow for future closed loop programs that modify output settings locally within the SRM.

V. PERFORMANCE AND TIMING

Figure 3 shows a logic analyzer display of the timing characteristics of a Local Station and its three Smart Rack Monitors. In response to the 15 Hz interrupt, the Local Station broadcasts an ARCNET message to the SRMs to initiate data collection. After 6.8 ms, the SRMs are ready to return their data. At 11.6 ms into the 66 ms cycle, data from all three SRMs has been received and processed by the Local Station. For this test, all SRMs were reading the onboard 64 A-D channels, the 8 bytes of digital I/O, and the readback of 16 D-A settings. Because the SRM activity is overlapped, an additional SRM would add only the ARCNET serial time plus the Local Station processing time, about 1.1 ms per SRM.

VI. PRESENT STATUS AND DISCUSSION

The Linac Upgrade is to be carried out in two parts; the replacement of the existing control system scheduled for January 1992, and the upgrade of the Linac itself, scheduled for late 1992. The SRM was prototyped and then fabricated in quantity by an outside vendor. Sixty-five SRMs have been delivered.

The concept of a networked data acquisition chassis that satisfies most of the interface requirements of the accelerator seems to work well. A single interface design that is duplicated in quantity results in a cost effective solution to the distributed data acquisition problem. Because of the single daisy-chained coaxial cable ARCNET connection, the SRMs can be installed wherever the data sources are located. If new devices are installed, another SRM can be easily added to read and control the additional equipment.

Data collection by SRMs is efficient because all SRMs collect their data in parallel. The VMEbus computer broadcasts a single command over ARCNET to initiate the data acquisition cycle and then processes the SRM messages as they are received.

Because it is a widely used network, ARCNET enjoys considerable industry support. A commercial copper- to-fiber optic ARCNET bridge is used to connect the VMEbus ARCNET adapter to the SRMs in the two 750 KV equipment domes of the preaccelerators. SRMs in the domes have the ARCNET copper transceivers replaced with pin-compatible fiber optic transmitter/receivers.

The connectivity provided by ARCNET allows future tasks to be handled by the SRMs. Because of the peer protocol of the network, one SRM could ask for and receive data from other SRMs. The other SRMs can be connected to the same or different Local Stations. Using this feature, an embedded control loop could be run in one SRM with data being collected from several other SRMs.

VII. ACKNOWLEDGMENTS

Several people contributed to the hardware development of this project. In particular, Robert Florian has been in charge of the VMEbus systems, and the early SRM design was done by Alan Jones, now at the SSC laboratory.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC02

Multi-processor Network Implementations in Multibus II and VME

Charlie BRIEGEL, Fermilab†
P.O. Box 500, Batavia, IL 60510, USA

Abstract

ACNET (Fermilab Accelerator Controls Network), a proprietary network protocol, is implemented in a multi-processor configuration for both Multibus II and VME. The implementations are contrasted by the bus protocol and software design goals.

The Multibus II implementation provides for multiple processors running a duplicate set of tasks on each processor. For a network connected task, messages are distributed by a network round-robin scheduler. Further, messages can be stopped, continued, or re-routed for each task by user-callable commands.

The VME implementation provides for multiple processors running one task across all processors. The process can either be fixed to a particular processor or dynamically allocated to an available processor depending on the scheduling algorithm of the multi-processing operating system.

I. INTRODUCTION

The motivation for a multi-processor platform in Fermilab's Accelerator Controls Group was to support our extensive commitment to CAMAC. The goal was to provide a replacement for PDP11 front-ends improve the effective utilization of the CAMAC serial link. Since CAMAC can have only one master, the link hardware allows ownership to be passed between cooperating processors. One of the requirements for this configuration was to implement ACNET communications for several duplicate processors running identical set of tasks. Thus, processors can be transparently added to the configuration with a corresponding increase in performance. This requirement provided the impetus for a multi-processor network connection to the existing network. The VME implementation provides support of multi-processor networks by using MTOS-UX MP (multi-processor version) operating system for transparent distribution of tasks among a set of processors.

II. ACNET OVERVIEW

ACNET (Accelerator Controls Network) is Fermilab's proprietary network protocol implemented in 1980. It consists of both a software protocol and a calling sequence specification.

The software protocol consists of a 9 word header (figure 1) preceding each message and provides a specification for the routing of messages between tasks. The protocol maintains a connection between cooperating tasks through a status reply and/or cancel messages. These notifications enable tasks and the network to maintain connectivity and cleanup network resources with minimal overhead.

The calling sequence provides a consistent user interface across heterogeneous machines and enables a request/reply paradigm for communication. Both asynchronous communications (traps, signals, or event flags) and synchronous communications (polling, wait, or wait with time-out) are supported by the calling sequence.

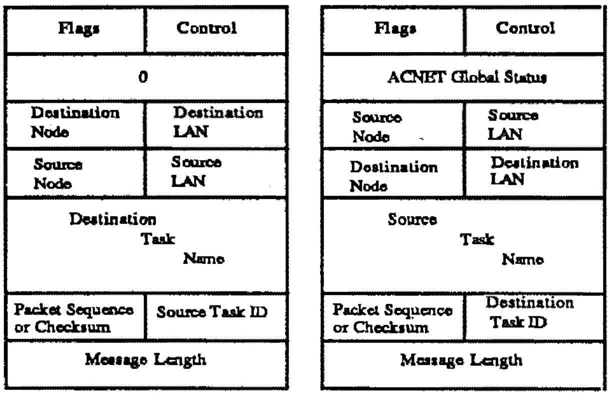
This calling sequence has enabled Fermilab to isolate effects to users due to changes in either the software or hardware protocol. While the implementation imposes inherent software costs, there is an advantage in providing a consistent layered approach for other software protocols.

The proprietary protocol does not restrict the use of standard protocols. For instance, by tunneling or encapsulating software protocols, ACNET has been implemented through a DECNET protocol and will be implemented with TCP/IP in the near future. The primary reason for not implementing a standard protocol stack has been the lack of support by vendors for the current set of heterogeneous processors and operating systems at Fermilab.

---

† *Operated by Universities Research Association for the Department of Energy.*

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC02



Flags	Control
0	
Destination Node	Destination LAN
Source Node	Source LAN
Destination Task Name	
Packet Sequence or Checksum	Source Task ID
Message Length	

Request/Unsolicited Msg Header

Flags	Control
ACNET Global Status	
Source Node	Source LAN
Destination Node	Destination LAN
Source Task Name	
Packet Sequence or Checksum	Destination Task ID
Message Length	

Reply Msg Header

Figure 1

III. MULTIBUS II

A. Configuration

The Multibus II bus with multiple Intel 386s running MTOS-UX SP (single processor version of the operating system) is the platform for this implementation. The crate contains an IBM/PC as the bus monitor which supports diagnostics, rebooting, and console emulation. Each of the Intel 386 processors has a daughter board supporting the CAMAC link directly.

Included in the configuration is a Token Ring board for communications. The board has an 8 MHz Intel 186 with a Token Ring daughter board attached to the ADMA controller of the processor board. The board implements the TMS380C16/04 chipset at either 4 or 16 Mbps and will be commercially available in the near future.

B. Software Implementation

To effectively communicate across a Multibus II backplane, each board contains a message passing co-processor and a software protocol called "transport" to provide a messaging facility similar to a LAN protocol. The Token Ring board's software takes an incoming transport message and synchronously transmits it onto the Token Ring. Upon receiving a Token Ring frame, a transport message containing the frame is sent to the appropriate processor. The distribution of a message is based on a connected task name for a given processor, the node number, and the message type.

ACNET requests are delivered in round-robin order. When a task called 'TSKA' connects to the network from CPU1, a transport message provides this information to the Token Ring processor. Another processor, CPU2, can connect with the same task name, 'TSKA'. Since replies must go to the originating request's processor, only requests are distributed in this way. The ACNET header's source node is altered to remember the specific processor which initiated the request so the reply is routed correctly.

ACNET header contains a 16-bit word to specify a lan/node number for both the destination and the source of the message. For this implementation, a set of node numbers are used as a logical nodes. One node number is used as a global logical node and implies the request can be distributed to any of the available processors which have the connected task specified in the ACNET header. For each processor, a specific logical node enables a request to be directed to the corresponding processor.

When a task is terminated, a cancel message is sent to delete each outstanding request. Since the requests are distributed in round-robin order, the Token Ring processor needs to deliver this cancel to the same processor which received the original request. Since cancels are infrequent and can be uniquely matched to original request through the ACNET header, the Token Ring processor broadcasts the message to all processors.

The byte orientation on the Token Ring wire was dictated by early implementations on VME, UNIBUS and QBUS platforms. The TMS380 chipset enables either Motorola or Intel logic interface which implies correct text, while integers are byte swapped. These early controllers used the Motorola interface and mapped the Big Endian format directly to the little endian processor bus. The result was an automatic byte swap. While this appeared to optimize our message format, the Multibus II Token Ring board using an Intel interface must byte swap each message.

The ACNET software for Multibus II and VME processors is written in "C" and is ported between the two platforms. The portable code is conditionally compiled to distinguish the use of transport as a vehicle to and from the Token Ring board.

C. ACNET extensions

Three new calls were added to aid in flow control. The round-robin scheduling of requests could be made ineffective if requests with greater life-times funneled into the same processor. The processor could be starved by servicing these requests with multiple replies while other processors would be relatively idle.

If such a case can be detected by the user, a NTSTOP call can be executed which provides a temporary block for future requests to be delivered to this task's processor. A subsequent NTUSTP call will resume normal round-robin schedule.

Since Multibus II implements a flexible messaging facility, a request can be re-routed to another processor based

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC02

on a user statistic, memory utilization, or idle time. The user can execute the NTMREQ call which will re-route the message to a specified processor. It is the user's responsibility to prevent a message storm on the bus.

IV. VME

A. Configuration

The test configuration used two Motorola 68020s running MTOS-UX MP operating system in a VME backplane. The platform implements one of the following three Token Ring boards: a Fermilab designed board at 4 Mbps, a Proteon p1542 at 4 Mbps, or a Formation fv1600 at 4/16 Mbps. All of the boards use the TMS380 family of Token Ring chipsets.

B. Software Implementation

The same ACNET software package implemented for MTOS-UX SP is used for the MP version. The operating system allows multiple tasks to run concurrently and transparently on multiple processors. A copy of the operating system is placed on each processor to improve performance. The operating system allows the user to specify the processor a task must run on or whether the task can be globally run on any of the available processors. If the task is global, the user must make modifiable data accessible to all processors. In the case of the test configuration, the operating system's tables, global task's data, and stack was contained in a shared memory segment accessed through the VME bus. Thus, performance is decreased due to the multi-processing operating system's overhead and each global task's data access over the VME bus.

V. PERFORMANCE

In a multi-tasking operating system, many factors determine the performance of the network to deliver messages to a task for useful work. Each of these implementations provide adequate performance with several opportunities for improvements. The Multibus II configuration can deliver 200 byte messages at approximately 110 messages per second for one processor and 180 messages per second for two processors. This measurement is for a task communicating with itself and in the case of two processors the messages are distributed to two duplicate tasks on separate processors.

The VME configuration was tested with one global task initiating a 200 byte request to a separate global task in the same VME backplane. The receiver of the request then sent an echoed reply of equal length. A single processor running MTOS-UX SP generated approximately 200 messages per second. A single processor running MTOS-UX MP generated approximately 160 messages per second. A dual processor running MTOS-UX MP generated approximately 170 messages per second. As expected, the additional processors should not improve network performance in such an architecture, but could provide additional compute power with ACNET communications.

The VME test was implemented with the ACNET network task running on the first processor. The same test was attempted with ACNET running as a global task on two processors. The result generated approximately 8 messages per second. The second processor had the Token Ring interrupts masked off. I believe the tasks were thrashing from processor to processor, but interrupt delivery to a task could only be delivered to the same processor which received the hardware interrupt. While further investigation is required, the results caution the user of MP to understand the system architecture or unusual results could occur.

Improvements in performance can be implemented with the following techniques:

1. Implement multiple Token Ring controllers.
2. Upgrade processors.
3. Overlap Token Ring I/O with transport I/O.
4. Eliminate unnecessary copy of received frames.
5. Optimize C code.
6. Upgrade the Token Ring to 16 Mbps.
7. Upgrade the TMS380C16/04 firmware.

VI. CONCLUSIONS

The Multibus II implementation has been running at Fermilab's Central Detector Facility. While the system has limited utilization compared to Fermilab Accelerator Controls, it has been valuable for detecting bugs and predicting performance characteristics. In particular, the round-robin scheduling with user invoked flow control appears to be adequate and manageable.

The throughput of the Token Ring board is a concern. Since the Token Ring software must byte swap each message, much of the performance is tied to the processor. The board will soon be available with a 16 MHz Intel 186 and should improve performance. Further, the software to enable multiple Token Ring boards on the same Multibus II backplane is easily implemented. Unfortunately, the ability to allocate these resources for incoming messages is a management problem without a stable solution.

Currently, their are no multi-processor implementations of MTOS-UX MP. While the operating system provides a high degree of transparency, the improvement in performance is difficult to quantify. In general, a master/slave or co-processor configuration with simple mailboxes is more predictable and manageable for real-time processing.

While both implementations use MTOS-UX as its operating system, the multi-processor version is inefficient for Multibus II. The operating system's global tables implies a time consuming access in Multibus II. The Multibus II implementation does enable tasks to be statically distributed across multiple processors. A VME implementation to enable duplicate tasks distributed across multiple processors requires mutual exclusion techniques on network structures and additional software. Thus, these two multi-processor network implementations provides a non-competitive solution solving two different problems.

VII. ACKNOWLEDGEMENTS

Multibus II Token Ring board was designed and built by Jim Zagel, John Smolucha, and Bob Marquart. Transport protocol for the Intel 186 was developed by John Smolucha and transport protocol for MTOS-UX was developed by Mike Glass and Bill Marsh. Kevin McGuire and Duane Voy (SSC) generated MTOS-UX MP for the MVME133 board.

VIII. REFERENCES

[1] Briegel, C., Johnson G., and Winterowd, L. (1989). The Fermilab ACNET Upgrade, Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems, pp. 235-238

[2] Glass, M., Zagel., J., Smith, P., Marsh, W., and Smolucha, J., (1989). The Upgraded Tevatron Front End, Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems, pp. 87-92

A DISTRIBUTED DESIGN FOR MONITORING, LOGGING, AND REPLAYING DEVICE READINGS AT LAMPF *

Mary Burns
Los Alamos National Laboratory
Los Alamos, NM, USA, 87545

Abstract

As control of the Los Alamos Meson Physics linear accelerator and Proton Storage Ring moves to a more distributed system, it has been necessary to redesign the software which monitors, logs, and replays device readings throughout the facility. The new design allows devices to be monitored and their readings logged locally on a network of computers. Control of the monitoring and logging process is available throughout the network from user interfaces which communicate via remote procedure calls with server processes running on each node which monitors and records device readings. Similarly, the logged data can be replayed from anywhere on the network.

Two major requirements influencing the final design were the need to reduce the load on the CPU of the control machines, and the need for much faster replay of the logged device readings.

I. INTRODUCTION

The Los Alamos Meson Physics Facility (LAMPF) linear accelerator, the injection lines, the experimental areas, and the Proton Storage Ring (PSR) contain over 16,000 beam line devices (approximately 11,000 in the LAMPF Control System (LCS) and 5,000 in the PSR control system) which can be accessed through the LCS Data System [1]. These devices include both monitoring and control devices which occur throughout the facility. Many of the devices are accessed through remote microVAXes which are connected by Ethernet to form a DECnet LAN with the main control computer, a VAX 8650. This system is becoming more distributed as remote computers are added to allow access to related groups of devices, and some of the CPU load is being off-loaded from the main computer to remote machines. The increased distribution of the system has made it necessary to redesign some of the software to take advantage of the changes. One such system is the software that monitors, logs and replays readings from devices located throughout the facility. Before the redesign of the software was started, a complete re-evaluation of the requirements of the software was carried out so that other improvements could be included in the new design.

II. FUNCTIONALITY OF THE SOFTWARE

2.1 The Sampling Software

A large number of beam line devices can be sampled periodically for the purpose of monitoring the device to determine if its reading is within a given range, logging the device reading to a history file, or both. The frequency at which each device is sampled can vary from once a second to once every eight hours. If a device which is being monitored is found to have a reading outside its given range (ie. it is out of tolerance) the accelerator operators are notified via the operator console and an error log. An alarm action may also be initiated. If the device reading is being logged to a history file for replay the reading is recorded together with a device identifier, a compressed timestamp, and the status of the read.

2.2 The Replay Software

The device readings can be replayed for a selected time period and displayed as graphs. Readings from several devices can also be combined in an arithmetic expression and plotted.

III. THE SOFTWARE DESIGN

The sampling software is divided into two parts; the program which collects the data and monitors and logs it, and the operator interface software which allows the operators to interact with the sampling program. Interactions include stopping and starting the sampling of individual devices or lists of related devices, changing the parameters associated with a device, such as the frequency of sampling, and requesting an immediate monitoring of a set of devices.

3.1 Problems with the Original Design

Originally the sampling software was designed to run on the main control computer and to monitor and log data from throughout the facility in a large, central, history file. Replay of device readings was carried out on the same computer.

As more nodes were added to the network the demand for distributed replay grew. Also the number and frequency of device sampling increased, so that the load on the control computer became more noticeable causing a degradation in the speed of execution of other applications, and affecting the ability of the facility operators to respond quickly to requests from experimenters or to emergencies. Due to the increasing size of the history file, replay of the device readings was also

---

* Work supported by the U.S. Department of Energy

becoming slower and consuming too much of the control computer's CPU and the operators' time.

3.2 An Interim Solution

One way in which replay has been improved is by the development of command procedures which run a version of the replay program as batch jobs at regular intervals. Instead of outputting the data to a graphics screen the device data is written to a file which is then sent to a laser printer. This automates the replay request process, and allows the replay to be scheduled at times of lower CPU usage so that there is less impact on beam production. However this is only a partial solution.

3.2 Requirements for the New Software

The requirements for the redesigned software include the ability to control the sampling of device readings from a number of nodes on the network, and the ability to replay device readings from any node on the network. It was also necessary to reduce the CPU load on the control computer for acquiring and logging device readings, and to increase the speed of replay of the device readings. The last two requirements were met by redesigning the history file. The software redesign was accomplished in two steps. The first step was to develop a distributed system, and the second was to redesign the manner in which the history file is shared between the logging and the replay software.

3.3 The Redesigned Distributed Software

As many of the devices are read through remote nodes it was decided that there are advantages in storing device readings on the local node to reduce network traffic and the load on the main control computer. The design of the software that interfaces to the sampling program calls for a user interface which can be run anywhere on the network. The user interface communicates across the network with servers running on each node which is sampling device readings. Communication between the operator interface and the server is done via remote procedure calls (RPC).

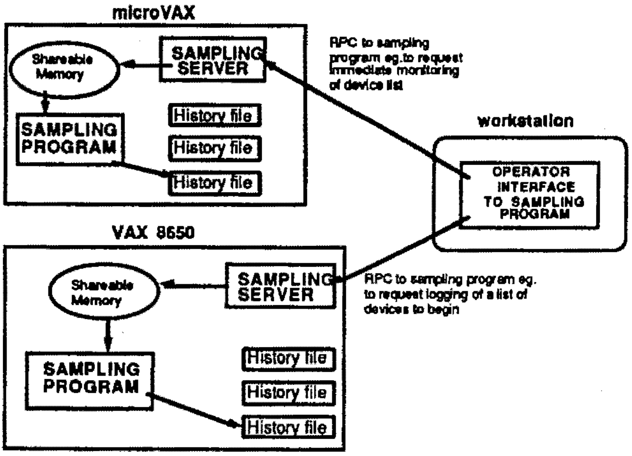


Figure 1. Diagram showing commands being sent to sampling programs on several network nodes, from a remote workstation.

An example of the way in which the distributed system of sampling software is used is shown in Figure 1. Sampling programs running on a microVAX and a VAX 8650 can be controlled from a workstation connected to the same network. Using the operator interface to the sampling program on the workstation the microVAX is selected and a request to immediately monitor a list of devices is send to the server on the microVAX. The server sends the request via shareable memory to the sampling program which carries out the action. The VAX 8650 is then selected using the same operator interface, and a request is sent via the server to the sampling program running on the VAX 8650 to start logging the readings of a list of devices to the local history file.

The replay software is also divided into a user interface which can be run from any node on the network, and a server running on each node which has recorded device readings at some time during the production cycle. Communication between the user interface and the server is again done with RPCs.

For example, a user on the VAX 8650 can request device readings from a remote microVAX. The data is retrieved from one or more of the local history files, converted to the appropriate units and returned to the VAX 8650 which then plots the data on a graphics screen (Figure 2).

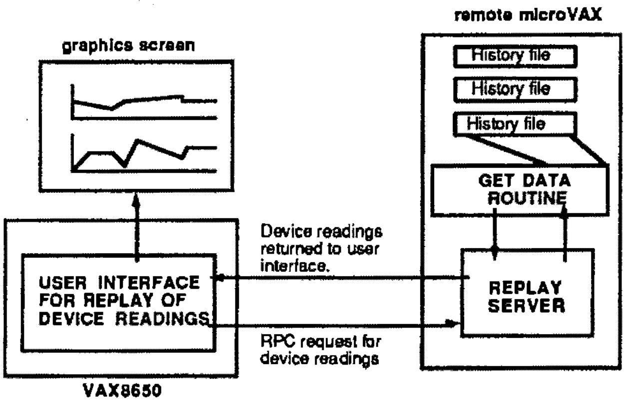


Figure 2. Method of retrieving data across the network and displaying it on a local screen.

3.4 Changes in the Structure of the History File

Originally device readings were recorded to a single file for the complete run cycle, which was typically five to six months, each year. In 1990, this file reached a size of about 300 Mbytes (ie.25,000,000 device readings). The file was sequential with direct access to the nearest eight hour pointer. Shared access to the file for both the logging program and the replay program was achieved with VMS locks, which resulted in both programs being slowed by lock manipulation. Another cause of slow data retrieval for replay was that each record read resulted in disk I/O.

The redesigned logging software opens a new history file every twenty four hours on each node where device readings are

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC03

being logged. When data replay is requested the replay software maps the relevant section of the history file into process memory and extracts data for the required devices. As replay of the most recent data must be available it may be necessary for the replay software to map to the section of the file being updated by the logging program. To prevent conflict between the two sets of software the history file is sequential and new records are appended to the file.

This method of sharing access to the file has significantly improved the rate at which data can be replayed because it has decreased disk I/O and does not require any file locking. It has also decreased the CPU usage for logging device readings as the file structure has been simplified and the need for disk I/O to access an intermediate time index file has been eliminated. Another benefit of the smaller history files is that they are easier to backup.

3.5 Other Design Changes

A high frequency of data sampling (every one or ten seconds) may be required for one or more devices over a period to study some effect or diagnose a problem. At LAMPF one device being logged every one second accumulates more than 1.3 Mbyte (86,400 device readings) of data per day. This frequency uses CPU during the logging process, and adds significantly to the volume of data stored in the history files so that it slows data retrieval for replay. To avoid unnecessary sampling at higher frequencies, the new software will allow the user to specifically enter the time period for which devices are required to be read at the one or ten second frequency, before they drop back to a less frequent sampling time. If no specific time is entered a default time will be used.

IV. FURTHER POSSIBLE IMPROVEMENTS

One possibility which has yet to be explored is whether the device readings should be grouped into seperate 24 hour files based on some relationship such as type of device, or location in the beam line. This would increase the CPU usage at the time the device readings were logged but the improvement in replay speed achieved by the further granularity of the history files could be significant. Changing to a different history file organization, such as an index file is not possible while the retrieval software is accessing the history file by mapping it into process memory. Another disadvantage of index files is that they are space consuming. At the moment it is possible to maintain the history files for a whole year on disk which allows for efficient replay of device readings over a complete production run.

V. PERFORMANCE GAINS

CPU usage by the new sampling software on the main control computer has been reduced by a factor of five but this is partly due to changes in the way in which the LCS Data System handles requests for device readings (message passing has been reduced by buffering device readings). It is hard to acquire accurate figures for the improvement in the speed of replaying device data, as this depends on the load on the computer and the size of the relevant history files. However for a short time both software systems were running on the same computer recording the same device readings in seperate history files. At that time it was possible to make accurate comparisons which showed that the speed of replaying data has improved by a factor of six. The average rate of data retrieval for replay is 200 to 300 readings per second on the main control computer. Replay of device readings from other nodes in the system is faster as history files are much smaller.

VI. PRODUCTIVITY GAINS

The sampling of device readings is now being carried out on several nodes on the network. In 1991 approximately 2,000 devices were sampled throughout the production run. Since the new software was implemented the amount of data logged has almost doubled to 40,000,000 device readings (ie. 650 Mbytes) for the 1991 production run. Replay of the device readings can be done anywhere on the the network.

VII. ACKNOWLEDGEMENTS

The author would like to thank Eric Bjorklund, Dave Schultz, and other members of the controls section of MP-6 for helpful ideas during the software design phase.

VIII. REFERENCES

[1] S. Schaller and E. Bjorklund, "Providing Common Data Access for the LAMPF and PSR Control Systems", Proc. 1987 IEEE Particle Accelerator Conference, Washington DC (IEE Publishing, New York, 1987) p 745.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC04

Synchronous Message-Based Communication For Distributed Heterogeneous Systems

N. Wilkinson
TRIUMF
4004 Wesbrook Mall
Vancouver, B.C
Canada V6T 2A3

D. Dohan
SSC
2550 Beckleymeade Avenue
Dallas, Texas
USA 75237

Abstract

The use of a synchronous, message-based real-time operating system (Unison) as the basis of transparent inter-process and inter-processor communication over VME-bus is described. The implementation of a synchronous, message-based protocol for network communication between heterogeneous systems is discussed. In particular, the design and implementation of a message-based session layer over a virtual circuit transport layer protocol using UDP/IP is described. Inter-process communication is achieved via a message-based semantic which is portable by virtue of its ease of implementation in other operating system environments. Protocol performance for network communication among heterogeneous architectures is presented, including VMS, Unix, Mach and Unison.

1 Introduction

The use of domain-driven object modeling techniques in the specification of the KAON Factory Control System[1] was in contrast to the more traditional emphasis on implementation and technology details and the consequent imposition of the technology on the requirements. When these object-oriented methods were used to model the KAON Factory Control System and to allocate the requirements specification to processor units[2], a *logical* architecture was derived which consisted of a network of distributed processors connected by two specialized communications buses: the *control* bus, a fast communication link responsible for the deterministic transport of control information and the *data* bus, a wide bandwidth link responsible for non-deterministic, data-intensive communication.

Since no generic processor platform can economically perform all of the functions required, the distributed network of computing platforms utilized by the KAON Factory Central Control System (KF CCS) will be non-homogeneous and will undoubtedly consist of both real-time and non real-time platforms. It is therefore important that a consistent software architecture be employed to implement communication among these platforms.

2 Message Based Architecture

The message-based semantic is a candidate for implementing the transparent, high performance inter-process and inter-processor communication required for the KF CCS. *Message passing* can elegantly encapsulate both task synchronization and data transfer into a small set of simple primitives having well-defined semantics[3]. Since a message header can identify a particular *method* to be used by a task *instance*, the use of the message-based semantic provides a convenient means of implementing an object-oriented architecture in a distributed environment.

In the *synchronous* message-based semantic, three primitives are employed for inter-process communication and synchronization: *send()*, *receive()* and *reply()*.

The *send* primitive implements the dispatch of a message to a destination task followed by the receipt of a reply from that task. Once it has called the *send* primitive, a task remains blocked until the receipt of a reply from the destination task.

The *receive* primitive is used to receive messages from other tasks and, typically, to wait for signals from interrupt service routines. Tasks which use the *receive* primitive remain blocked until a message or signal arrives, or until a user-determined timeout occurs. The use of the *receive* primitive for the receipt of both messages and signals at a single point of execution considerably simplifies the structure and design of tasks which must simultaneously deal with external events and communicate with other tasks.

The *reply* primitive is non-blocking. It is used by tasks, such as servers, which cannot block while dispatching a message. A *reply* is always made to a task which has used the *send* primitive to send a message to a given task and is waiting for that task to reply.

Task synchronization is implicit in the communications primitives employed. For instance, a *server* object should never block when posting a message to another task. For this reason, servers employ the *receive* primitive to receive messages from clients and the *reply* primitive to respond to clients. *Courier* objects are used to carry messages between servers, as a server would block if it employed the *send* primitive to communicate directly with another server.

A synchronous message-based semantic may be constructed from the more primitive inter-process communications semantics offered by operating systems such as VMS or VxWorks, or it may be obtained as the native semantic

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC04

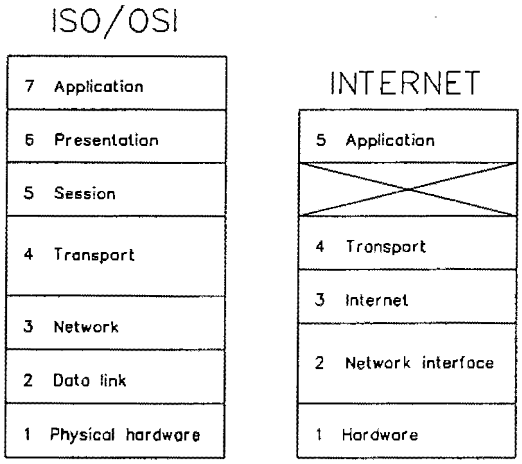


Figure 1: Protocol Stacks

of message-based operating systems, such as the Harmony real-time kernel. For instance, the Unison real-time operating system from Multiprocessor Toolsmiths implements the synchronous message-based semantic using the native semantics of the Reliant kernel, a commercial variant of Harmony, or as a messaging layer added to the pSOS+ real-time kernel.

3 Network Architecture

Common to all modern network protocols is the separation of *logical* elements of the protocol into several *layers*. The five layer Internet protocol stack and the seven layer ISO/OSI protocol stack are illustrated in Figure 1.

Proponents of the Internet suite of protocols, which has been widely implemented in the Unix and other domains, claim that it is the de-facto network standard. In the realm of networked real-time operating systems, the Internet protocol suite is almost exclusively used.

3.1 Potential Implementations

Ethernet, the IBM token ring and emerging technologies such as the Fibre Distributed Data Interface (FDDI), are prime candidates for implementation of *control* and/or *data* bus architectures. FDDI, with its high bandwidth and deterministic response, could perform both *control* and *data* bus functions. In some situations, provided that its bandwidth and non-deterministic response is acceptable, ethernet could be used for both functions at a considerably lower cost.

3.2 Transport Layer Issues

The Internet *virtual circuit* transport layer protocol, TCP, employs a *sliding window* protocol. When used for real-time control on an unreliable network, this protocol may exacerbate non-deterministic response due to the retransmission of unnecessary data segments. In addition, since a sliding window protocol transmits a number of segments before waiting for an acknowledge, unacceptable time delays may be introduced in waiting for urgent control information to be acknowledged. For this reason, the more rudimentary UDP transport protocol, which implements the unreliable transfer of datagrams, is preferable. However, the use of UDP requires additional transport layer functionality in order to ensure the reliable transfer of datagrams.

3.3 Session/Presentation Layer Issues

As shown on Figure 1, the Internet protocol stack has no session layer. For the reasons described in Section 2, a session layer employing a synchronous message-based semantic is considered desirable.

The Internet protocol stack also lacks a presentation layer. For a message-based session layer, the critical presentation layer service is one which ensures that the representation of data is the same regardless of the platform employed. The use of such a service ensures that a distributed network of heterogeneous processors can understand each other's messages. The *External Data Representation* (XDR), which has been designated RFB1014 by the ARPA Network Information Center, provides the required presentation layer functionality.

4 The Socket Server

During the KAON Factory Project Definition Study, a session layer protocol for network communication was developed which implements network communication using a synchronous, message-based semantic. In addition, a reliable virtual circuit transport layer which does not employ a multiple segment sliding window protocol was developed over UDP. This protocol suite is known collectively as the *socket server*[4] and consists of three co-operating objects which communicate via a synchronous message-based semantic.

The session layer of the socket server is embodied in the *socket_server* task and a reliable virtual circuit transport layer is embodied in multiple paired instances of *socket_task* and *recvfrom_task*. Socket server clients use the *ss_open* primitive to open a network connection and the *ss_close* primitive to close a network connection. Once a connection is open, socket server clients communicate with each other via the synchronous, message-passing primitives *ss_send*, *ss_receive* and *ss_reply*.

The socket server has been successfully implemented on Unison, VMS, Mach and SunOS[1] platforms. Ethernet message-passing performance between a Unison platform and VMS, NeXt/Mach and SunOS platforms has been measured. Socket server ports to RISC platforms are forthcoming.

5 Performance Evaluation

The hardware configuration of Figure 2 has been used to evaluate socket server performance. The transceiver fan-out was used to connect two or more systems for performance measurements. The VMEbus Unison platform consisted of a 25 MHz Motorola 68030 processor board and a

---

[1] Sun 3/60

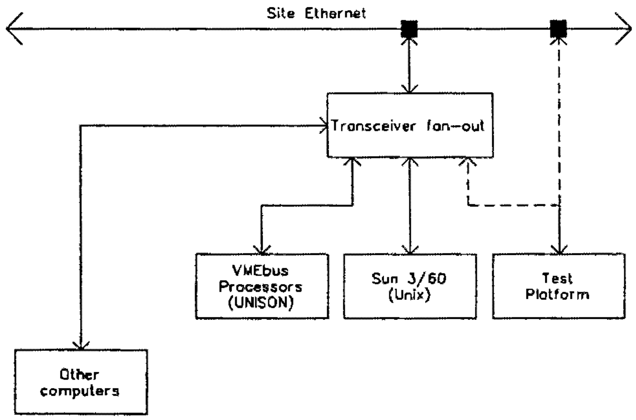


Figure 2: Equipment Block Diagram

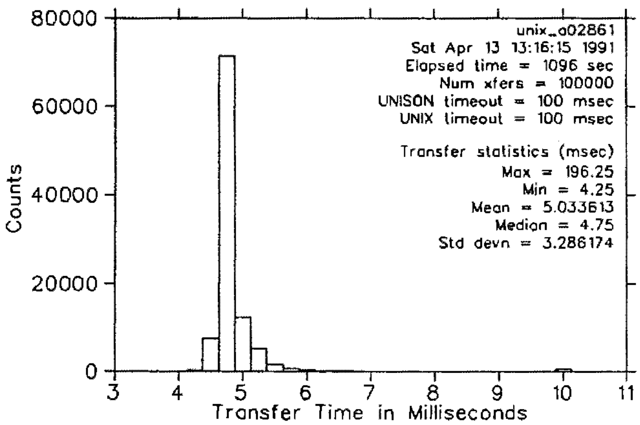


Figure 3: Four Byte Message Transfer Times

low-cost ethernet processor equipped with a 10 MHz Motorola 68010 processor. The Sun 3/60 served as the software development host for the Unison system and was the first test platform for the socket server performance.

Round-trip message transfer times were measured by the Unison client, which communicated with the non-Unison client using the ss_send primitive. The Unison client also employed ss_send to periodically transfer the acquired data to the non-Unison client for storage in a file. The one-way message transfer time was taken as half of the round-trip time. Figure 3 illustrates a typical histogram of one-way message transfer times obtained during performance tests for a Sun 3/60.[2]

For the purpose of performance measurement, only the Unison platform used the native synchronous messaging primitives for inter-process communication among the socket server tasks. The VMS, Mach and SunOS platforms were evaluated using a single-threaded variant of the socket server which emulated message passing using message copying and function calls.

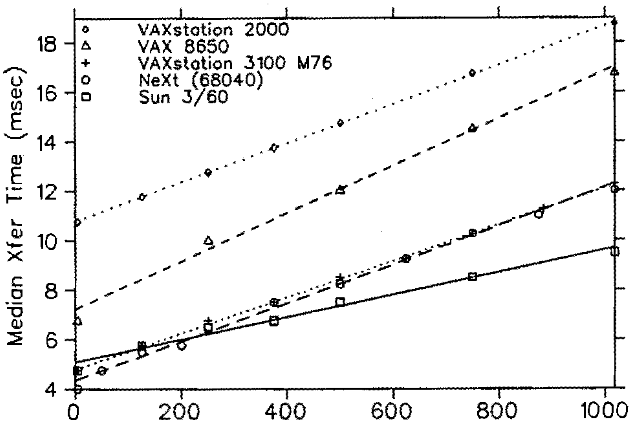


Figure 4: Socket Server Performance

6 Results

Figure 4 presents graphs of message size versus median transfer time for each configuration tested.[3] For each set of points, the data was fitted to a line, which is also shown on Figure 4. Since the Unison platform was common to all measurements, Figure 4 shows the relative performance of the non-Unison platforms tested. Figure 4 indicates that the NeXt platform obtained the best performance for small message sizes. However, for message sizes greater than about 200 bytes, the Sun 3/60 performance was superior to all others. The VAXstation 3100 M76 and the NeXt platform were quite similar in performance, despite the large difference in their costs.

As only the round-trip transfer time was measured, it is impossible to separate the Unison and non-Unison protocol overheads. However, the round-trip transfer time was recently measured between the Sun 3/60 platform and a high performance Unison platform using a single 25 MHz Motorola 68030 processor with on-board ethernet. The median transfer time between these platforms showed an improvement of only 0.75 milliseconds as compared to the transfer time measured between the Sun 3/60 and the two-processor Unison platform which was employed for the rest of the tests. This suggests that the low performance ethernet processor in the two-processor Unison platform does not make a significant contribution to protocol overheads. The Unison overheads will be established by a forthcoming measurement of the round-trip transfer times between two identical Unison platforms.

Table 1 presents the median message transfer rates for each platform tested. The NeXt achieved the highest message transfer rate of 250 messages/second for 4 byte messages. An ethernet implementation of the *control* bus would be limited to this maximum message transfer rate for present day CISC processors. However, a performance improvement is anticipated upon completion of the forthcoming socket server port to RISC platforms

---

[2] The 10 millisecond bin of Figure 3 contains the sum of all counts for which the one-way message transfer time was 10 milliseconds or longer.

[3] The median transfer time has been chosen as it provides a better indication of the peak on the corresponding histogram.

Platform	Data transfer rate (messages/second)	
	4 byte messages	1018 byte messages
VAXstation 2000	93	53
VAX 8650	148	60
VAXstation 3100 M76	211	82
NeXt	250	83
Sun 3/60	211	105

Table 1: Median Message Transfer Rates

Platform	Data transfer rate (kbytes/second)	
	4 byte messages	1018 byte messages
VAXstation 2000	0.36	53
VAX 8650	0.58	59
VAXstation 3100 M76	0.82	81
NeXt	0.98	83
Sun 3/60	0.82	105

Table 2: Median Data Transfer Rates

(SPARCstation, DECstation). Although ethernet is non-deterministic, the histogram of Figure 3 is sharply peaked at the median transfer time. Very few one-way transfers required more than 6 milliseconds and the maximum transfer time was 196 milliseconds. Provided that the ethernet bandwidth is adequate and that ethernet saturation can be avoided, this level of determinism may be acceptable for some *control* bus applications.

Table 2 presents the median data transfer rates obtained for each platform. For a 1018 byte message, the data transfer rate for the Sun 3/60 platform was in excess of 100 kbytes/second. If ethernet were employed for the *data* bus, the maximum data rate would be of the order of 100 kbytes/second for CISC platforms. It is expected that superior data transfer performance would be obtained if the socket server employed a multiple segment sliding window protocol (TCP) for *data* bus transfers and reserved the non-sliding window protocol for *control* bus applications.

Since it is suspected that the largest contribution to socket server protocol overheads is made by the non-Unison processor, the forthcoming socket server port to RISC platforms may considerably enhance socket server ethernet performance. However, FDDI is the prime candidate for truly high performance *control* and *data* bus applications.[4] Due to its redundant counter-rotating token ring scheme, FDDI is both reliable and deterministic. Since the FDDI standard incorporates a layered software and hardware architecture, it will easily accommodate change and platform heterogeneity.

7 Conclusion

The socket server is well suited for communication among distributed heterogeneous systems. For instance, the synchronous message-based semantic has recently been ported to VMS platforms and this enables a multi-process VMS socket server. Since the socket server uses Unison messaging primitives, it will function in both the Unison/Reliant and the Unison/pSOS+ environments. A socket server port to the VxWorks real-time operating system is also anticipated.

If the protocol overheads of a layered network implementation are unacceptable and present day technology must be employed, the KF CCS *control* and *data* buses must be implemented using custom/proprietary systems. Such systems may be networked systems using custom hardware and software, they may be distributed shared memory systems or they may be hybrid shared memory/networked systems. In general, such systems are more awkward to implement and expensive to maintain because they do not easily accommodate change. In addition, typical custom/proprietary systems do not easily support an operational environment which consists of heterogeneous computing platforms.

The ethernet socket server is suitable for *control* and *data* bus applications provided that its limitations (250 messages/second, 100 kbytes/second, non-deterministic) are acceptable. If these limitations are not acceptable then RISC/FDDI socket server ports or custom systems are required.

References

[1] C Inwood, G A Ludgate, D A Dohan, E A Osberg, and S Koscielniak. "Domain-driven specification techniques simplify the analysis of requirements for the KAON factory central control system". *Nuclear Instruments and Methods in Physics Research*, A293(1,2):390–393, 1990.

[2] E A Osberg, G A Ludgate, S Koscielniak, and D A Dohan. "Dynamic object modelling as applied to the KAON control system". *Nuclear Instruments and Methods in Physics Research*, A293(1,2):394–401, 1990.

[3] Peter J M Baker. "A programming paradigm based on the send-receive-reply task communication primitives". Master's thesis, Department of Electrical and Computer Engineering, University of Victoria, Victoria, B.C., Canada, 1988.

[4] Neil A. Wilkinson. "Message-based network communication between heterogeneous systems". KAON Factory Project Definition Study TRI-DN-91-K164, TRIUMF, 4004 Wesbrook Mall, Vancouver, B.C., Canada V6T 2A3, June 1991.

---

[4] A Unison TCP/IP and UDP/IP port to FDDI is anticipated by Fourth Quarter this year.

The Transmission of Accelerator Timing Information around CERN

C.G. Beetham, K. Kohler, C.R.C.B. Parker and J.-B. Ribes

European Organization for Nuclear Research (CERN)

CH - 1211 Geneva 23, Switzerland

Abstract

Prior to the construction of the Large Electron Positron (LEP) collider, machine timing information was transmitted around CERN's accelerators using a labyrinth of dedicated copper wires. However, at an early stage in the design of the LEP control system, it was decided to use an integrated communication system based on Time Division Multiplex (TDM) techniques. Therefore it was considered appropriate to use this facility to transmit timing information over long distances. This note describes the overall system, with emphasis placed on the connectivity requirements for the CCITT G.703 series of recommendations. In addition the methods used for error detection and correction, and also for redundancy, are described. The cost implications of using such a TDM based system are also analyzed. Finally the performance and reliability obtained by using this approach are discussed.

I. INTRODUCTION

In the planning phase of the LEP collider it was recognized that the much greater physical size of LEP compared with previous accelerators would result in a radical change in the way communications systems of all types would be implemented. The larger size implied correspondingly longer cables and more signal regenerators, and consequently much more expense. Moreover, most cables had to be routed through the LEP tunnel itself, significantly limiting the space available for future exploitation of the tunnel. It was evident that the number of cables would have to be reduced by multiplexing information for several different users onto one cable. Consequently, it was decided to install a multiplexed system throughout LEP as a general user facility.

II. ACCELERATOR TIMING SYSTEMS

A. General Features

Typically, computer control systems have a real-time response in the order of 10 to >1000 ms. Whilst this is adequate for many applications there is always a need to trigger selective hardware equipment with a finer time resolution. This is achieved by means of a timing system.

An accelerator timing system is simply a fast broadcast message transfer system. The messages (events) are normally short (typically less than 32 bits) and should have a maximum time resolution of 1 ms. The jitter of each message is determined by the transmission medium and by the sampling processes at the generator and receiver. Depending upon the type of machine, the events are either pre-programmed, according to the specific machine cycle, or triggered by external stimuli, such as emergency beam dump.

A principle difference between a timing system and a control system is that when a transmission error is detected, the control system normally retransmits the message. This is inappropriate with a timing system as, inherently, the message contains a timing reference.

Although the timing and control systems may be considered as two separate identities they are in fact strongly coupled to the machine that they are controlling. For this reason timing systems have always been designed in-house and tailored to the specific accelerator concerned. This implies producing timing equipment compatible with the chosen control system. Equipment of this type is normally not available commercially.

For a large accelerator a typical timing system consists of three parts:

- a central timing generator
- a long distance transmission network
- a local distribution system.

The timing system chosen for LEP and subsequently used on the SPS has been described elsewhere[1]. This article concentrates on the long distance transmission network.

B. Long Distance Transmission

In the case of CERN's Super Proton Synchrotron (SPS) machine, "long distance" refers to the distance between two adjacent auxiliary surface buildings located above the SPS tunnel. For the LEP machine it refers to the distance between two adjacent tunnel alcoves. For both scenarios the distance is between one and two kilometers.

For the original SPS machine the accelerator timing information consisted of a 1 ms clock and short (7 bit) trigger messages called events. The events were Manchester encoded and transmitted to each building over a video cable containing two good quality twisted pairs. One pair was used to transmit the 1ms clock whilst the other carried the events. The transmission rate was 333 kbit/s and no error detection or correction was employed. "Tap-off" amplifiers were installed in each building in order to transmit the timing signals around the SPS. These were eventually duplicated in order to reduce the downtime of the machine.

In addition a separate twisted pair cable was used to transmit the revolution frequency (43kHz). Identical amplifiers were used in each building; again they were duplicated. This latter system is still operational, whilst the SPS general machine timing has since been upgraded to the LEP system.

C. *The Timing Distribution Standard*

The local distribution of the timing information is done according to EIA specification RS-485. The data format adopted is supported by integrated circuits from National Semiconductor (the NS8342/8343 transmitter/receiver set). These ICs frame the 32 bit event into four bytes. Each frame is enveloped within a predefined start/end sequence and in addition each byte starts with a synchronizing bit and ends with a parity bit. The NS8343 receiver chip contains a seven bit error register which indicates the detection of a mid-bit transition fault, an invalid ending sequence and also a parity error. The contents of this register are used for error detection.

The entire frame is Manchester encoded. This coding scheme was developed at Manchester University in the U.K. for data recording onto magnetic media. The principle characteristic of the code is that no D.C. component is transferred to the transmitting line medium. This is achieved by ensuring that each data bit has an equal negative and positive component, as shown in Fig. 1.

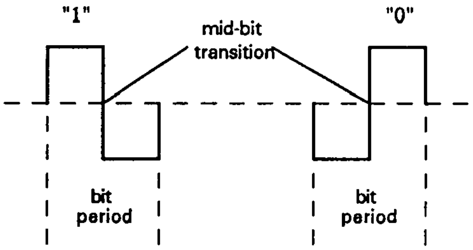


Fig. 1 Manchester Encoding

The use of Manchester coding doubles the bandwidth requirement because each bit period must contain a mid-bit transition.

III. MULTIPLEXING SYSTEMS

There are two main methods of multiplexing: frequency-division multiplexing (FDM) and time-division multiplexing (TDM). FDM uses a different frequency band for each signal; analogue filters are used to extract the required signal at an access point. These filters require tuned circuits which do not lend themselves to large-scale integration, which require careful setting up, and which are prone to drift. This, together with the general trend towards digital transmission, led to the decision to use time-division multiplexing for LEP.

In time-division multiplexing a number of bits are taken sequentially from each of the multiplexor's input ports and applied to the output port. The converse operation is applied in the other direction. This procedure is shown, simplified, in Fig. 2.

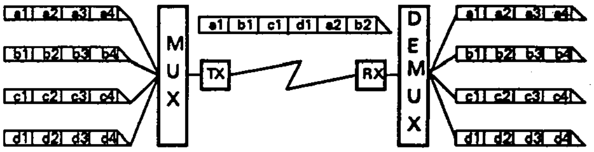


Fig. 2 Simplified TDM System

In practice TDM systems are far more complex. This is because the high-rate data streams carry extra information, particularly for synchronization purposes and in order to deal with clock rates which are only nominally synchronous.

TDM equipment is used widely around the world by PTTs and other telecommunications suppliers and users. The standards for the equipment have been set by the CCITT worldwide [4]. Since compatible equipment is available from a large number of vendors, the market is very competitive and therefore the prices are low.

As previously mentioned, it was decided to adopt TDM technology for LEP [2]. This is used to transmit the machine timing and also the revolution frequency timing information. The latter system has been fully described elsewhere [3].

The timing events are transmitted at 512kbit/s over a 2.048Mbit/s TDM link. At the 2.048Mbit/s data rate the G.703 Recommendation specifies a coding scheme known as HDB3. The TDM-Timing interface unit makes this code conversion transparent to users of the timing system.

The interface standard (CCITT Recommendation G.703) for these high speed connections is being used more and more frequently. For example, video CODECS, LAN bridges, and terminal concentrators are available with G.703 interfaces. However, one disadvantage of using a standardized approach to interfacing is that, in special situations some interface conversion is required. As will be shown later the cost of this conversion, in the LEP situation, is outweighed by the economies brought about by multiplexing.

In the case of the interface between the timing system and the TDM network it was decided to subcontract to industry the design and production.

IV. DESCRIPTION OF THE TDM-TIMING INTERFACE

Fig. 3 shows a block diagram of the TDM-Timing interface.

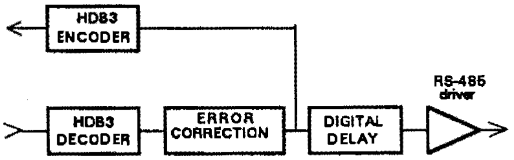


Fig. 3 TDM-Timing Interface

The interface unit was specified at CERN but designed and manufactured externally. The four principle parts of the interface are described below.

A. *HDB3 Decoder/Encoder*

The incoming 2.048 Mbit/s data stream is first galvanically isolated and terminated in a 75 Ohm resistive load. Thereafter, it is passed on to the HDB3 decoder. The HDB3 code is somewhat similar to Manchester coding, in that it has a low DC component and also contains clock information, but it is much more efficient in bandwidth usage. The HDB3 decoder generates a NRZ data signal and associated clock. In addition the decoder generates two alarm signals: "loss of input" which is activated when the input is no longer present, and "violation" which is generated when violations of the HDB3 coding law are detected. Both of these signals are monitored by the control system.

The HDB3 encoder simply performs the opposite function to the decoder, ie. it converts the NRZ signal to HDB3 code and passes it on to the outgoing transformer isolated driver.

B. *Error Correction*

It can be seen that Manchester coding inherently introduced some redundancy, as does the superimposition of each Manchester bit on the TDM links. In effect four 2.048 Mbit/s TDM bits are being used to convey one 512 kbit/s timing bit.

At the specification stage of the TDM-timing interface it was decided to exploit this redundancy and incorporate a simple error correction system. Four TDM bits are taken at a time and applied to the error correction circuitry. The four possibilities for single-bit errors when transmitting Manchester codes are indicated in Table 1.

"1"		"0"	
Correct code:	1100	Correct code:	0011
Incorrect code:	0100	Incorrect code:	1011
"	1000	"	0111
"	1110	"	0001
"	1101	"	0010

Table 1 Single-bit Errors

As can be seen from the above Table, in the event of a single bit error introduced by the TDM network there is still enough information to distinguish between a 1 and 0. In fact the scheme can also correct some double-bit errors introduced by the TDM network. As this network has a very low error rate ($< 10^{-12}$) this means that the error rate seen by the timing system is virtually zero.

C. *Transmission Delay Compensation*

The SPS/LEP timing system was designed so that any specified event would arrive at all of the related equipment around the machines with a one millisecond resolution encompassing a maximum jitter of two microseconds. However in physically large accelerators such as the SPS and LEP there are significant propagation and equipment delays. In order to compensate for this, delays were incorporated in the interface unit. Each delay consists of a 12 bit register incremented by the 2.048MHz. clock derived from the incoming data stream. The required value can be written and read by the control system.

D. *Local Timing Output*

The output of the timing interface conforms fully to the electrical characteristics defined by CCITT Recommendations V.11 and X.27 and EIA specifications RS-485. The differential signal is Manchester encoded.

V. REDUNDANCY

Redundancy, or more specifically duplication, has been installed following invaluable experience gained from the original SPS timing system installation.

The TDM links provided for the SPS and LEP timing systems have been configured in a double ring topology, one ring for each machine. The links on each segment of the ring are inherently full duplex; this feature has been exploited to provide two independently routed signals, following clockwise and anticlockwise routes around each ring. These alternate signals are used in the interface equipment for added reliability.

Besides profiting from the full duplex feature of the TDM, the interface units were also duplicated as well as the power supplies used to power the local line drivers. Switching to the redundant system is done automatically when necessary. In addition the selection can be made via the control system. The equipment is constructed in Euromechanics as defined in DIN 41914.

VI. COST CONSIDERATIONS

It is clear that the cost benefits brought about by the use of multiplexing will be greater as the number of channels on a link increases. In the case described here each link carries three channels: 2Mbit/s for the general machine timing, 2Mbit/s for the beam synchronous timing, and 8Mbit/s for the machine control token ring.

A careful analysis of the cost of the installation has been done, as well as an estimate of the cost involved in implementing the transmission of these three channels in the traditional way on dedicated links. It was found that these two costs were roughly equal, ie. in this case there has been, as yet, no significant cost advantage in multiplexing. However, there is considerable spare capacity to each of the 24 locations served by the network, and this allows extra channels to be put into service very quickly and cheaply, as no significant additional cabling is required. This flexibility has already proved extremely useful.

VII. CONCLUSIONS

Although in terms of bandwidth, the timing system is not a major user of the TDM system it is a vital user, ie. if the TDM is not operational, the timing system cannot work. The use of the TDM network for the transmission of timing information has proved to be more reliable than the previous transmission system which used dedicated cables and repeaters.

The error-correction features are extremely effective: since the introduction of this system there has been no lost beam time due to transmission errors. The cost of supplying the extra bandwidth for error correction is far outweighed by the advantages of improved reliability, and error correction would be an indispensable part of any future system. In fact studies of advanced error correction algorithms are presently under way with a view to using these techniques for a pilot implementation of an integrated fast control and timing system [5].

VIII. ACKNOWLEDGEMENTS

Many people have contributed to the success of the project to transmit timing information in the manner described in this paper. Particular mention must be made, however, of B. Amacker and P. Nouchi.

IX. REFERENCES

[1] C.G. Beetham, R.J. Lauckner, C.G. Saltmarsh, Overview of the SPS/LEP Fast Broadcast Message Timing System, Proc. of the 1987 Particle Accelerator Conference, Washington, D.C., Vol. 1, p. 766.

[2] C.R.C.B Parker, The Provision of Telecommunications Links for the LEP Collider, 1989 Particle Accelerator Conference, Chicago, Ill.

[3] G. Baribaud et al., The Beam Synchronous Timing System for the LEP Instrumentation, International Conference on Accelerator and Large Experimental Physics Control Systems, Vancouver, Canada 1989.

[4] CCITT "Red Book", Fascicle III.3, International Telecommunications Union, Geneva 1985.

[5] C.R.C.B. Parker, M.E. Allen, H.C. Lue, C.G. Saltmarsh, A New Data Communication Strategy for Accelerator Control Systems, International Conference on Accelerator and Large Experimental Physics Control Systems, Vancouver, Canada 1989.

Time and Load Measuring in the SPS/LEP Control System

J. Navratil
Czech Technical University
Zikova 4, 166 35 Praha 6, Czechoslovakia

Abstract

This paper describes the experiences with the SPS/LEP Control System during its first operational days from the communication point of view. The results show difference between hardware possibility of the local communication based on the modern technology and the possibility to use it by PC machines. There is also several figures describing the activity on the communication lines.

I. INTRODUCTION

The control system of the SPS/LEP is based on distributed microcomputer network which contains more then 300 hosts and many thousands of different devices used for the control. Backbone is created by the communication system which can be divided into two parts. The horizontal plane is based on the Token Ring system to which are connected hosts used for control of particular processes on high level and many vertical planes provide connection onto the control devices (computers, interfaces, VME creates, etc.). MIL-1553 B is used here as the standard for the communications.

Such control system is very interesting first of all by its simple design where the total computer power is obtained by the relatively small quantities of hosts (IBM PC clones) and also for the sophisticated combination of a different communications devices (TDM, gateways, bridges, etc.) which ensure flexible services in the large area and long distances (27 km) [1][2].

My activity was concentrated on the Token Ring services and because my programs had to communicate through the whole network it was interesting to know the environment in which I had to work and its basic characteristics as the response time, speed of data transport and the load of the system (from the point of view of communication). Detail results are reported at CERN SL/CO/Note 90-05.

II. NETWORK ACTIVITY

During November and December 1989 when LEP started normal operation and all parts of the control system were working normally I collected some data about the activity of all hosts in the Token Ring communication network of the Control system. The aim of the measurement was to get a general overview of how much the Token Ring network was used by users for real control work.

For this purposes a set of programs were developed which are part of the Network Management System [2]. The results and evaluation of the activity in the network was used mainly to get an overall picture about the use of particular parts of the network and for long term planning.

For described experiment I have used three programs. The first of them collects data from the selected hosts and two others are used for processing and analysing the data and printing reports. The interface counters are used as the basic information. (The counters of the number of transmitted packets and bytes are a part of the standard communication software.) Data is collected by the RPC mechanism by the standard Rply_data program which is running on almost all hosts as a standard server. All the information from the selected ring can be collected in a few seconds.

The analysing program generate 3 tables. The first gives the total overview about the measured values, the second table gives an extract of the most active hosts and tries to express the values in a pseudo graphic form. The third table shows the activity in the time picture (the same data was collected in regular time intervals).

If we have a look on the tables in detail, then the first table contains 5 originally measured values (time in sec., number of input packets, number of output packets, number of input bytes and number of output bytes) for each host. All other values in the table are calculated from these values. As the best representation of the activity of each host I have used the percentage of the total activity in the network. These figures were calculated for each measured value (input packets, output packets, input bytes, output bytes). The last calculated value for each host is the "average speed", it is the activity of the communication interface - sum of the input and output bytes divided by measured time. On the bottom of the table are summary values and an "actual average speed" in the ring.

The second and third table are self-explanatory. An extract of the hosts in the second table is done on the basis of the minimum trash value. In our case the trash value was selected as 1.5%. From the ghraphic interpretation it is immediate to get a picture of the hosts activity (the most used one). The third table shows all hosts but for better reading of the information there are shown only relative values of these hosts in which the activity is higher than the trash value. This table gives a good overview about the "long term behaviour" or the "stability of needed service". (See examples shown in tables.)

III. MEASUREMENTS OF RESPONSE TIME

There are several tools for the network communication but I have used only two of them, FTP or (TFTP) and RPC (Remote Procedure Call). Both tools ar also heavily used in many others applications. My effort has been concentrated on the RPC implemented via Network compiler [5] from the point of view of the normal user. This is because this facility was the principal tool for many programs in the Network Management System which we are developing in our section and also because RPC is a very powerful tool in itself.

Some timing measurement were done in the past. For FTP and transport of short messages by the UDP facility (which artificially simulates the possibilities of real RPC [4]), the work was mainly concentrated on analysing the communications properties of different operating systems. Another study [6] measured response time of an implemented RPC, but

Table 1 An example of distribution of network load
(time interval 4450 sec, trach level 1.5 %,
40 machines in the PCR ring)

Input	**packets**			**Output**	**packets**	
host	**abs.**	**rel.[%]**		**abs.**	**rel.[%]**	
aldev5	5431	12.14	XXXXX	5096	13.02	XXXXX
cons11	1568	3.51	X	1673	4.27	X
cons14	3107	6.95	XXX	2928	7.48	XXX
fspcr	22739	53.30	XXXXXXXXXXXXX	227239	58.11	XXXXXXXXXXXXXX
pipcrl	2192	4.9	XX	2033	5.19	XXX
rfsba3	2263	5.06	XXX	2060	5.26	XX

Table 2 An example of hosts activity
(scan interval 360 sec., trash level 1.0 %)

Scan interval/Activity [%]

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
host									
aldev5	10.9	22.5	22.5	33.4	22.7	32.7	7.1	4.8	6.5
aldis1	.	.	.	.	.	.	.	.	.
aldis2	1.4	10.7	.	.	.	.	.	.	.
colsep	.	.	.	.	.	.	.	.	.
cons11	2.8	6.2	6.6	7.5	5.8	8.4	1.9	5.8	1.7
cons14	42.9	3.8	2.6	1.3	20.3	3.5	2.5	.	2.7
fspcr	35.0	29.6	47.8	35.6	28.2	31.8	83.6	54.2	84.8
pipcrl	4.0	9.8	10.1	10.6	9.9	13.7	2.5	2.3	2.5
rfsba3	.	.	.	.	.	.	.	26.2	.
...									

at the start of my work I didn't know about this paper. The results from this study are for similar conditions approximately the same, the main difference is in the method of measuring. While the measurements described in [6] took many hours, my method takes only a few seconds and the load of the measured object was minimal. It allows these measurements to be made practically on-line without interrupting normal operations in the ring or in the host. Another new effect which has not been mentioned in [6] is a problem with ring interconnections. This is mentioned in [4] but there are only figures for a one packet pass. From the point of view of a normal user of RPC this is very important because in normal circumstances the effect increases with each packet and it is also directly connected with the global topology of the network.

Random timing measurement have shown that the response time is sometimes quite high and more dependent on the hardware of the computers which are involved in the communication than on other known aspect such as position of the host in the ring, distance or speed of the transmission system ! The results of RPC response time from one host to several others in the network are shown in table 3.

Explaining this effect is rather simple. The token ring is based on the IBM TR system with frequency slightly modified to 4,225 MHz. And if we assume that the physical level of the TR system is capable of transferring data at about 4.0 Mbps [3] then the time used for the real physical transmission of data used in the RPC (< 1 kbyte) are in the microseconds range which is negligible compared with the time consumed by all communication software layers implemented under Unix (including the RPC layers).

Timing measurement were done by a trivial client program which was running in the selected computers and a standard program which was running everywhere as a permanent server.

The results from many measurements showed that the faster reply came if both source and destination were fast computers. The following measurements showed the interesting fact that the same result was possible to get if both the Client and the Server program run in the same computer. This allowed simplification of the whole measurement and to define a Round Trip Response Time - RTRT which could be taken as a

Table 3 RPC Response time from the PCR ring

from			**to**	
ring	**(host)**	**ring**	**(host)**	**response**
pcr	aldev5	lma	lmagrlma	170 ms
pcr	aldev5	lma	bile28	190 ms
pcr	aldev5	lsv	lmgrlsv	170 ms
pcr	aldev5	lsv	colsr2	220 ms
pcr	aldev5	laa	lmgrlaa	170 ms
pcr	aldev5	laa	recpca	200 ms
pcr	aldev5	lbar	aldev4	210 ms
pcr	aldev5	ldev	olive	250 ms

reference characteristic for a particular type of machine. The values include time for connection. This time was about 80 – 100 ms for the I80286 processors and 60 ms for the I80386 machines (Olivetti M380/C).

All previous measurements were taken in one simple ring. Working outside simple rings brings another aspect to be taken into account. The rings are linked together by bridges and gateways. These devices (driven by software) need some time for doing their job. It was surprising to find that the delay in the bridges is, from the point of view of this measurement, negligible but the delay caused by the gateways is rather high. The values for different gateways on the network were measured for our type of request as being between 15 to 50 ms (worse for 80286 – 45 ms and the best for CISCVO/DATATRAC – 15 ms). It is necessary to emphasize that these times represented an average case from a not very loaded gateway. These figures can be much worse and they will be quite heavily dependent on the traffic through a particular gateway. Knowing the characteristic of the particular constitutive elements of the system and also knowing the topology of the network we could relatively easily synthetically estimate the best response time for the different paths in the network.

Finally I would like to bring to attention several other important problems which are connected with the use of the UNIX operating system and which put a little more light on the results.

The first problem is the Task Management. All results which I have presented here were directed towards the communication area and therefore I always had to use the shortest response time (best case from many measurements). That means that they were taken after the server process was firstly activated. The response time of the first request in which the activation (fork) is done contains another delay and therefore the first RPC response is much longer than the RTRT presented in previous tables. In practice it represents a time between 0.3 – 6 sec. depending on the load of the host. For the user it means that he must calculate this fact and if he needs fast response must keep his process in permanent activity instead of starting it repeatedly.

A second problem is more general. Performing short accurate software timing measurements within UNIX is practically impossible. There is normally a timer running with microseconds accuracy, however the system doesn't update it's software image fast enough. There is a 20 ms timing quantum which I couldn't overcome. This limitation was very annoying mainly in cases of very fast processors and this fact could have an influence on the precision of the measurements.

IV. CONCLUSION

On the basis of these results and knowledge of the network topology we can estimate the response time in any part of the network. This can be useful for future decisions or for reconfiguring the system because other applications will have similar response time overheads. The measurement also showed that the communication system has enough capacity to transfer substantially higher load through the network and also shows that the bottle-neck of the communication system is not in the hardware level of the TR but rather in the software or hardware of the hosts and gateways.

V. ACKNOWLEDGEMENT

The results of that work has been made possible by the close collaboration within the team. I would like to thanks particularly P. Lienard, L. Guerrero and A. Bland for their contribution to this work and P. G. Innocenti and K. Kostro for fruitful discussions.

VI. REFERENCES

[1] P. G. Innocenti, "The LEP Control System", International Conference on Accelerator and Large Experimental Physics Control System, Vancouver, Canada, November 1989.
[2] P. Lienard et al., "The SPS and LEP Control Network Architecture", International Conference on Accelerator and Large Experimental Physics Control System, Vancouver, Canada, November 1989.
[3] J. S. Haugdahl, "Inside the Token Ring", North-Holland 1987.
[4] K. Kostro, "Host Performance in the TCP/IP Internet of SPS", Internal report, CERN-SPS/ACC, January 1989.
[5] P.S. Anderssen, V. Frammery, G. Morpurgo, "User guide to the Network Compiler Remote Procedure Call (NC/RPC), LEP Control Note 97, May 1989.
[6] P. Van Der Stock "Remote Procedure Call Measurements", CERN SPS/ACC Note 16/12/1988.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC07

THE ELETTRA FIELD HIGHWAY SYSTEM

D. Bulfone, P. Michelini, M. Mignacco
Sincrotrone Trieste, Padriciano 99, 34012 Trieste, Italy

Abstract

ELETTRA is a third generation Synchrotron Light Source under construction in Trieste (Italy); it consists of a full energy linac injector and a storage ring with beam energies between 1.5 and 2 GeV. The ELETTRA control system has a distributed architecture, hierarchically divided into three layers of computers; two network levels provide communication between the adjacent computer layers. The field highway adopted for the connection of the middle-layer local process computers with the bottom-layer equipment interface units is the MIL-1553B multidrop highway. This paper describes the hardware configuration and the main communication services developed on the MIL-1553B field highway for accelerator control. As an additional feature, typical LAN utilities have been added on top of the basic MIL-1553B communication software allowing remote login and file transfer; these tools are currently used for software development in our laboratory.

I. INTRODUCTION

The ELETTRA Control system is distributed over the about 260 m Storage Ring circumference and 170 m Linac-plus-Transfer Line, with an architecture based on three computer levels (presentation, processing and equipment interface) and two network layers.

High performance UNIX workstations with excellent graphical capabilities are used as operator consoles at the presentation level. The upper layer network, which is based on Ethernet and the TCP/IP communication protocol, connects the control room workstations and the distributed process level computers called Local Process Computers (LPC).

The LPCs, bridging the two network levels, are the "core" of the control system, where all the main application control tasks are executed, acquiring and processing data from the equipment interface level; the ELETTRA LPC consists of assemblies of VMEbus single-board computers (SBC), running the OS-9 operating system. In order to enhance both performances and modularity, a multiprocessor-multimaster architecture has been developed for the LPC, where each board is allowed to take VMEbus mastership and execute its own data transfers.

Separate lower level network branches connect each LPC to the equipment interfaces, called Equipment Interface Units (EIU). Following a definition widely accepted by the control system designers community [1], the term "field highway" is used to indicate the communication system between the local process computers and the interface-level processors.

The VMEbus standard and the OS-9 operating system are adopted also at the EIU level; the typical EIU configuration consists of one microprocessor board associated with several input/output boards of different type.

Exploiting the high performances of the field highway, we have assigned the LPCs with operational criteria, which enhance system design clarity at the same time: 2 LPCs and 48 EIUs are foreseen for magnet power supplies control, 1 LPC and 7 EIUs for vacuum, 1 LPC and 12 EIUs for storage ring beam position monitors, 2 LPCs and 3 EIUs for linac control, etc. A total number of 14 LPCs and 88 EIUs is to be finally installed.

II. THE MIL-1553B MULTIDROP HIGHWAY

In the project of an accelerator control system, the choice of the field highway is strategic: in spite of the growing of standards in the informatics and electronics world, many different solutions are currently used and proposed. The main parameters considered in the choice of the ELETTRA field highway are: communication topology, electrical noise immunity and data integrity, deterministic response, cost and performance. After a careful investigation of the non-proprietary commercially available products, we decided to adopt the MIL-1553B standard [2], slightly modified for accelerator control.

A. Communication topology

The MIL-1553B standard, originally developed for the aircrafts by the U.S. Department of Defense, defines a serial Time Division Multiplexing (TDM) highway on which one Bus Controller (BC) can communicate with up to thirty Remote Terminals (RT) in a multidrop configuration; the BC always provides data flow control and is the sole source of communication. This hierarchical scheme perfectly fits into our control system architecture: placing the BCs and the RTs at the LPC and EIU levels respectively, separate MIL-1553B branches connect each LPC to the EIUs it supervises. A typical configuration is shown in figure 1. Moreover, a multidrop highway topology together with the appropriate communication software permit to add and/or remove EIUs with no system shutdown, catering for future upgrades, requests or simple maintenance.

B. Electromagnetic noise immunity and data integrity

In order to guarantee a very good immunity from the electromagnetic noise of an accelerator environment, a shielded twisted-pair cable is adopted as MIL-1553B highway physical medium on which Manchester II biphase coded differential signals are transmitted.

In addition to that, the following intrinsic "acknowledged message" exchanging mechanism is used to assure data integrity: the BC sends a receive/transmit command word to the addressed RT, eventually followed by up to thirty-two data words; the RT receives the command, receives or transmits data as directed and responds with a status word. Parity checking is applied on each word boundary.

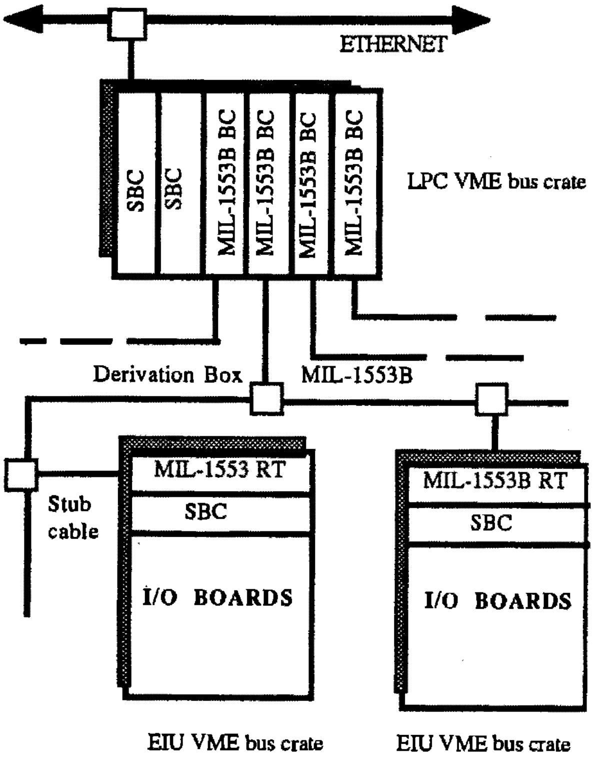


Figure 1. The ELETTRA field highway system.

C. Deterministic response

Dealing with the accelerator equipment at the interface level, the undefined network access times associated with the CSMA/CD (Carrier Sense Multiple Access with Collision Detection [3]) methods of Ethernet cannot be accepted; a deterministic behavior of the field highway is strongly requested. In the MIL-1553B standard configuration the BC "polls" (the so called "roll-call polling" type [3] is used) the status of its RTs at regular time intervals under software control and checks if new data is ready to be received. On this basis, real time operation with pre-configured timings is achievable.

D. Cost and performance

The price of the single highway interface board must be carefully considered, as a high number of nodes (about 100) are connected, especially at the RT level. Using commercial type electronics, the cost of an RT node is lower then that of a standard input/output interface board. The adopted physical medium is also inexpensive and no special installation tool is needed.

In order to provide the necessary flexibility of the control configurations implemented for ELETTRA, the field highway must be able to operate over distances up to some hundreds meters: with the chosen implementation, a raw bit transfer rate of 1 Mbit/s over about 300 m can be achieved; lower transmission speeds for longer distances are possible.

The MIL-1553B standard permits to send broadcast messages, which are often useful for accelerator control: issuing a special type command word, the BC can transmit to all the connected RTs at the same time.

III. THE ELETTRA FIELD HIGHWAY HARDWARE IMPLEMENTATION

The highway physical medium consists of a 100 % shielded twisted-pair cable which has a characteristic impedance Z_0 = 78 Ohm; each branch is matched by two termination resistors of value $Z_0 \pm 1\%$. The nominal attenuation, measured on a 1 MHz sinusoidal signal, is about 15 dB/Km; the wire-to-wire distributed nominal capacitance is 64.6 pF/m.

Each MIL-1553B device (BC or RT) connects with the highway through a so called derivation box and a short (less than 2 m) stub cable: the derivation box contains a couple of resistors which are serially inserted on the two stub cable wires in order to prevent BCs or RTs from damaging in case of short circuits on the highway.

Both the BCs and the RTs [4] are equipped with on-board transformers for ground isolation between nodes. They use a commercial type VLSI protocol chip automatically dealing with the "acknowledged message" exchanging mechanism described above.

The BCs have been integrated in the multiprocessor-multimaster environment of the ELETTRA LPC. Starting from a CERN/SL BC design [5], we developed a standard VMEbus Requester and implemented some hardware modifications [6] in order to achieve full compatibility with the other LPC commercial VMEbus SBCs; this allows future increases in the LPC performances as more powerful boards are available on the market. The OS-9 operating system is installed on each BC board.

IV. BASIC COMMUNICATION SOFTWARE

The communication software allows both MIL-1553B equipment and VMEbus LPC SBCs to exchange user messages.

The communication software for the field highway system mainly consists of three layers (figure 2): physical, translation and routing.

A. Physical layer

The physical layer is mainly in charge of shielding the low level communication details. Since many MIL-1553B boards can be both BC and RT, we decided to split this level into two parts; the first one is strictly related to the adopted hardware, while the second part handles the most common functions (circular buffers, fragmented packets, etc.). After defining a physical number for each board, user applications can easily exchange data packets through the network, independently of the hardware used. The physical number is fixed by on-board switches and it is not usually changed after the first setting.

The only drawback of this addressing mode is that identical boards with different numbers cannot be exchanged without modifying and recompiling all the user programs.

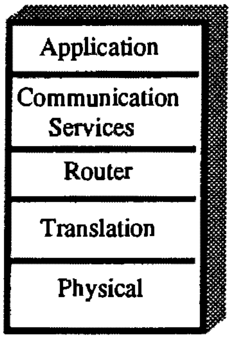


Figure 2. The ELETTRA field highway software layers.

B. Translation layer

The translation layer, built on top of the physical one, avoids the aforementioned problem. It consists of an OS-9 driver maintaining an internal table of correspondences between physical numbers and the new logical station numbers which are used by the routing layer. A logical station number is associated with each LPC board and with the SBC managing the EIU RT.

C. Routing layer

The routing layer, or router, performs a reliable user message transmission starting with a configuration file which describes the logical system topology. All the stations attached to the same physical medium (VMEbus or MIL-1553B) form a so called "subsystem" and the configuration file contains as many lines as the subsystem number.

Many advantages are associated with the layered architecture. The software is easily ported on different MIL-1553B boards; user messages between two stations attached to different subsystems can be exchanged independently of the interconnecting physical medium; the identical routing interface is maintained. In order to add new subsystems and stations, the following steps have to be taken:

- develop a new dedicated low level hardware driver and eventually a new address translation driver;
- assign new logical station numbers to each subsystem SBC;
- add the new subsystem station numbers to a new line of the routing configuration file.

V. THE ROUTING LAYER INTERFACE LIBRARY

A clear interface library to the routing layer functions is provided to allow a straightforward implementation of the higher level services. This interface consists of a collection of C language routines which have a common first argument: the communication channel number. This is an integer ranging from 0 to 7, which is associated with a routing path, a service code and a circular buffer for the incoming user messages. At present only one channel per process is used, but up to eight are available to increase flexibility.

The description of the principal library calls follows:

error = OpenRouterService (channel, bufnum, service)

int error;	router error code or zero
int channel;	channel number
int bufnum;	number of circular buffers
short service;	communication service code

OpenRouterService opens a communication channel with a specific service code;

error = CloseRouter (channel)

int error;	router error or zero
int channel;	channel number

CloseRouter closes the specific channel previously opened by OpenRouterService;

error = WriteMsg (channel, buffer, len, dest, service)

int error;	router error or zero
int channel;	channel number
char *buffer;	message buffer
int len;	message length
short dest;	destination logical station number
short service;	communication service code

WriteMsg transmits a message to the specific destination station in a reliable way;

error = WaitMsg (channel, timeout)

int error;	router error or zero
int channel;	channel number
in timeout;	timeout in system ticks, 0 for infinitum

WaitMsg waits for a user message to arrive. The type of service is fixed by the OpenRouterService function.

VI. COMMUNICATION SERVICES

The described communication interface is the base on which all the communication services are built. Even if it represents a flexible and hardware independent interface, it cannot be given to the application programmers for software development. A typical user is not, in fact, aware of the logical station numbers, circular buffers and communication service codes; he refers to the accelerator equipment only by logical names and uses the communication services without taking care of the control system topology details. In order to provide a completely symbolic equipment access [7] [8], we have developed a higher software level, which shields part of the router library function parameters.

The Symbolic Address Resolution, Command/Response, Inform, Alarm and Broadcast services have been written. Their implementation is based on the general concepts of client and server processes, with clients located on a LPC (Command/Response and Broadcast) or an EIU (Inform and Alarm) according to the different service types and data flow involved.

A. Symbolic Address Resolution service

This special service hides the incoming user message logical routing from the application programmer. It consists of a server process running on the LPC and an associated table called Equipment Directory Unit (EDU).

When a client request arrives, the server process looks up the logical equipment name in the EDU and translates it into a logical station number. Moreover, the server checks the equipment access permissions and provides for on-line EDU reconfiguration.

B. Command/Response service

The Command/Response service allows the user applications to access the equipment. There are two operating modes: synchronous and asynchronous. In the first case the client sends a command and suspends its execution until a result comes back from the server. In the asynchronous mode the client process sends a command without stopping and, after some time, reads the results prepared in the meantime by the server process.

A typical Command/Response request, including EDU access, takes about 30 ms.

C. Inform service

The Inform service has been developed in order to cater for the necessity of sending data from an EIU to the LPC without waiting for an LPC data request.

The Inform is therefore very similar to the Command/Response service: the main difference is that it allows only unidirectional data flow.

D. Alarm service

The surveillance programs have a fundamental role among the processes running on an EIU; they continuously check the equipment status and detect anomalies or fault conditions. When a serious error occurs, these programs must act as client processes sending an alarm message to the associated LPC server. In this special case we are interested in delivering an alarm message as fast as possible. In order to increase the transmission speed, the alarm message does not access the EDU table for symbolic destination address resolution.

E. Broadcast service

The Broadcast service permits to send the same user message to all the EIUs attached to the same MIL-1553B multidrop highway branch. Exploiting the MIL-1553B standard broadcast capability described above, we are studying the possibility of using this service to generate a software synchronization among the EIUs: the transmitted broadcast messages can in fact produce hardware interrupts on the EIUs, waking up the processes to synchronize.

VII. IMPLEMENTED LAN UTILITIES

Taking advantage of our layered software structure, we have developed a special level which interfaces the Translation Layer with the commercial OS-9/NET [9] communication software package (figure 3). OS-9/NET is based on the Network File Manager (NFM) and provides for typical Local Area Networks (LAN) utilities like remote login and homogeneous file access. These tools are the basis for the distributed software development environment we created in our laboratory: using a single LPC equipped with a disk, we can download software and start task execution in the diskless EIUs; moreover, the laboratory LPC works as a common remote disk server and virtual terminals can be opened on the EIUs for software testing.

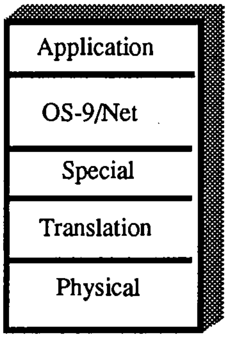


Figure 3. OS-9/Net layer implementation.

In the final system implementation, where diskless LPCs and EIUs are installed, NFM and TCP/IP tools can be effectively combined. In this way a remote login from the control room can be executed on both LPCs and EIUs; a UNIX disk server can also be shared by all the field highway system stations.

VIII. CONCLUSIONS

Starting from the military MIL-1553B communication standard, a multidrop field highway has been developed for the control of the ELETTRA equipment.

Despite the low cost of the used communication boards (BC and RT) and physical medium, the typical functionalities needed for accelerator control are catered for. Compared with existing similar products, the implemented MIL-1553B highway offers one of the best cost/performance ratio available.

The chosen VMEbus MIL-1553B BC board has been integrated in the multimaster environment of the ELETTRA LPC, increasing system performance at least by a factor of two.

The layered communication software allows to send data packets between the LPC SBCs and the dropped EIUs in a completely transparent way. A data transmission rate of 70 kbytes/s has been achieved with 100 byte packets.

The implemented LAN utilities are currently used for the development of system and application software.

IX. ACKNOWLEDGEMENTS

We are grateful to the CERN/SL control group who supplied the first MIL-1553B equipment and software, from which our development started. We thank Luca Barbina and Luca B. Giovannetti for their valuable technical support.

X. REFERENCES

[1] D.P. Gurd and M. Crowley-Milling, "Preface" in *Proceedings of the ICALEPCS*, Vancouver, BC, Canada, October 30-November 3, 1989.
[2] *MIL/STD-1553B, Aircraft Internal Time Division Command-Response Multiplex Data Bus*, Department of Defense, Washington, DC 20360, USA, 1978.
[3] A. S. Tanenbaum, *Computer Networks*, Prentice-Hall International Editions, 1988.
[4] D. Francart et al., "MIL-1553B Multidrop Bus for Controlling LEP and SPS Equipment", LEP Controls Note 56 Rev., August 1987.
[5] C. Guillaume and C. Pignard, private communication.
[6] L. Barbina and D. Bulfone, "The MIL-1553B Bus Controller Board for the ELETTRA VMEbus Multimaster Local Process Computer", Sincrotrone Trieste Internal Report ST/M-TN-91/3.
[7] "The LEP/SPS access to equipment", LEP Controls note 54 Rev., Geneva, March 1988.
[8] M.C. Crowley-Milling, "Naming Conventions for the ELETTRA Components", Sincrotrone Trieste Internal Report ST/M 88/13.
[9] *OS-9/NET Version 3.0*, Microware Systems Corporation.

Network Communication Libraries for the Next Control System of the KEK e^-/e^+ Linac

Norihiko Kamikubota, Kazuro Furukawa, Kazuo Nakahara and Isamu Abe
National Laboratory for High Energy Physics (KEK)
Oho, Tsukuba, Ibaraki 305, Japan

Abstract

The network communication libraries for the next control system of the KEK Linac have been developed. They are based on TCP/IP sockets, and show high availability among the different operating systems: UNIX, VAX/VMS, and MS-DOS. They also show high source portability of application programs among the different computer systems provided by various vendors. The performance and problems are presented in detail.

I. INTRODUCTION

The KEK 2.5-GeV electron/positron linac has been controlled with a distributed processor network since its first operation in 1982 [1,2]. Since, however, the system resources have become inadequate for increasing demand, we have introduced several subsystems in order to extend the system capability [3]. Furthermore, we have studied the possibility of system rejuvenation by a complete replacement of the minicomputers and their associated fiber-optic network with new ones. The proposed next control system comprises Unix-based workstations as a man-machine interface, an Ethernet as a high-speed communication network, and VME stations as front-end systems [4].

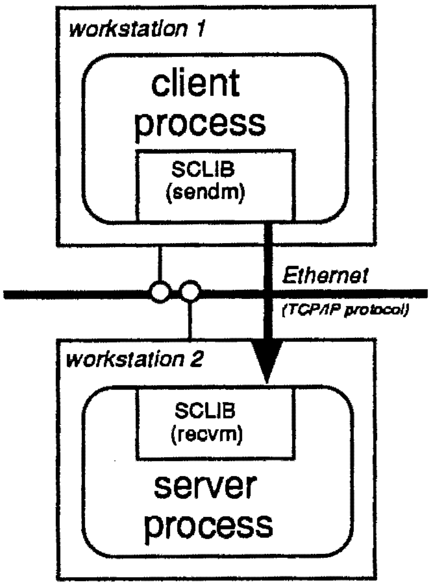
It is apparent that the tools for communication between workstations and VME stations are needed in the proposed system. In addition, some of the subsystems (the operator's console subsystem which comprises DOS-based personal computers [5], a diagnostic expert system for the linac injector developed in a Unix workstation [6], a beam-current monitor developed in a VAXstation [7], and so on) are expected to be used with the next control system. Thus, communication availability between different operating systems is important in our case.

We have developed a network communication library called "SCLIB" for the media of Ethernet with the TCP/IP protocol. Details are described in section II. Another library used to control the magnet power-supplies in the KEK linac, called "MGLIB", has been developed as an improved version of the SCLIB. The features are demonstrated in the section III. A discussion related to these libraries is presented in section IV. We hope that our present experience will provide good guidance for those who intend to introduce a similar network communication system.

II. NETWORK COMMUNICATION LIBRARY

A. Principles for the Library

The library "SCLIB" has been developed for the TCP/IP protocol. It was designed as a tool for real-time data transfer between processes, which is different from a file-transfer tool (like FTP), a file-sharing tool (like NFS), and an



```
main()  /* client process */
{
 int sc_open(), sendm(), sc_errend(), sc_close(); /* sclib */
 extern int sc_errno;  /* error code holder for sclib functions */

 sd = sc_open( "service_name@nodename" );  /* open connection */
  if( sc_errno < 0 ) sc_errend(); /* error message & exit */

 rtn = sendm( sd, "message" );       /* send a charcter string */
  if( sc_errno < 0 ) sc_errend();

 sc_close( sd );                     /* close connection */
}
```

Figure 1
Example of transferring a character string from a client process (workstation 1) to a server process (workstation 2). The basic flow of the SCLIB function calls is also shown.

Table 1
Computer systems used in the present performance tests. An operating system and a rough estimate of the CPU power for each computer system is also given.

Computer	Operating System	MIPS
DEC DECstation5000	ULTRIX	24
Sun SPARCstation1	SunOS	12
NeXT NeXT	Mach	10
Mitsubishi MX3000II	OS60/UMX (System-V based)	3
Fujitsu A-60	SXA (System-V based)	3
DEC VAXstationII/GPX	VMS WIN/TCP installed	1
NEC PC9801ES (16MHz,80386)	MS-DOS PCTCP installed	-

interactive communication tool (like telnet). The library provides C-language functions. A typical example is shown in figure 1, together with the basic flow of the SCLIB function calls.

The principles that we considered when we starting to develop the library are the following: The first principle is communication availability between different operating systems. We have adopted an inter-process communication technique called "stream socket", which is based on the TCP/IP protocol. It provides the basic network communication functions with error-handling schemes for C language. The socket is one of the standard inter-process communication methods in Unix-based workstations, and is also available on other computer systems.

Since the socket functions require various kinds of network parameters, it is almost impossible to use by those who are not familiar with this field. Thus, the second principle is to prepare easy-to-use functions for application programmers. As shown in figure 1, the SCLIB functions used in a client process require only two parameters: the destination of data ("service_name@nodename") and the data, itself ("message"). In addition, the existence of an error-handling routine (*sc_errend*) makes the client program very simple.

The third principle is to ensure high source portability of application programs among computer systems provided by various vendors. This leads to an easy replacement of the hardware of our present control system. Actually, we have found several "small" differences in the socket functions provided by different vendors, and have tried to include vendor-dependent parts within the library as many as possible. The use of the C language is also preferable from this viewpoint.

B. Communication Throughput

In order to study the overall communication throughput, including library overhead, test programs with the SCLIB functions have been prepared. We have carried out measurements of the data-transfer times between two computer

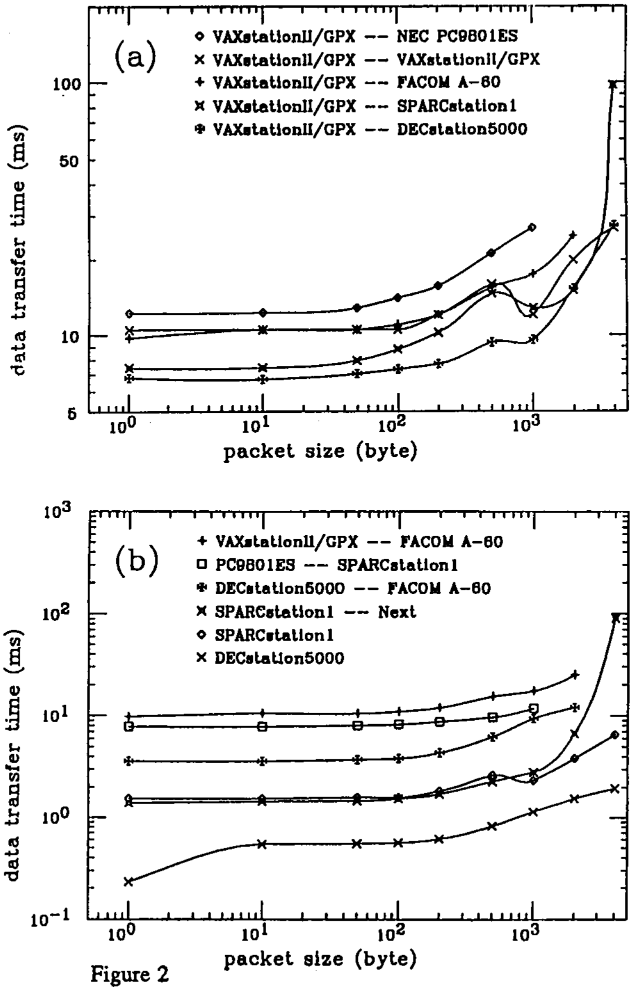


Figure 2
Measured data-transfer times between the computer systems listed in table 1.
(a) Data-transfer times between a VAXstationII/GPX and other computer systems.
(b) Data-transfer times for other combinations of computer systems. Two of them (SPARCstation1 and DECstation5000) are the result of the case that two processes are in the same workstation.

systems (processes) for the seven computer systems listed in table 1. The results for typical cases are shown in figure 2.

The following points can be pointed out.

(a) The data-transfer time takes 0.2-30 ms for 1-1000 bytes of data, depending on the CPU power of the computer systems used.

(b) The data-transfer time remains almost constant as the size increases from 1 to 100 bytes. It typically becomes twice as the size increases from 100 to 1000 bytes.

(c) The best record among combinations of computers in table 1 was found in the case that the two processes are at the same workstation, DECstation5000. The data-transfer time was 0.2-1 ms for the 1-1000 bytes data.

(d) Assuming a data size of 1 kB, the overall throughput is evaluated to be 400-1000 kB/s for data transfer between typical workstations (10 or more MIPS).

In addition, the dips observed at around 1000 bytes are considered to be caused by the buffering scheme of the TCP protocol.

C. Discussion

The round-trip response time between the operator and a local control device was studied in the present control system [1]. It was evaluated to be of the order of 100 ms for our data of 128 bytes. According to the results of our present measurements, one or two orders better throughput can be expected in the next control system. If workstations with greater CPU power (for example 40-100 MIPS) become available in the future, further improvement in the communication throughput can be expected.

The measured communication throughput seems to be sufficiently high for most of the applications required to control the accelerator. However, it should be noted that the time required to open network connection, which usually takes 0.1-5 seconds and is necessary each time to start inter-process communication, is not included in the values given here.

III. MAGNET CONTROL LIBRARY

A. Introduction

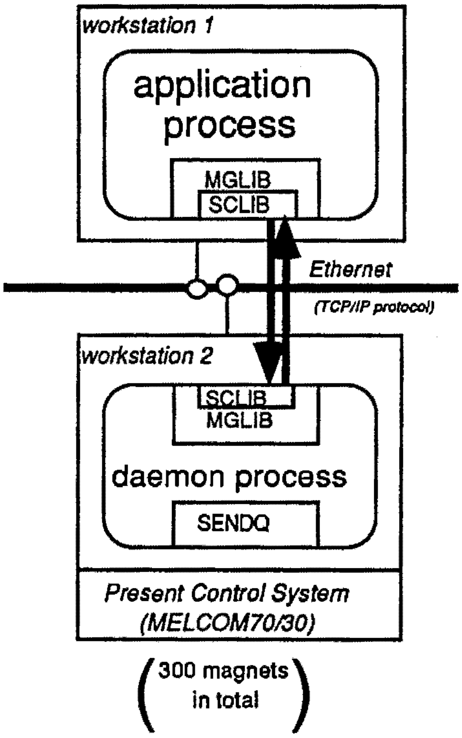
The development of applications used to control the magnet power-supplies of the KEK linac was possible only with the minicomputers used in the present control system (Mitsubishi MELCOM 70/30) and its backup workstation (Mitsubishi MX3000II). In order to realize a better environment for software development, we first tuned the workstation serving as a gateway between an Ethernet and the present control system [3]. We then developed a library called "MGLIB" in order to make it possible to control the power-supplies at any of workstations connected with the Ethernet.

The mechanism of how a power-supply is controlled is shown in figure 3. When an application process calls a MGLIB function, an information message is sent to the daemon process in the gateway. The daemon process controls the power-supplies according to the received message. It then returns a reply message which includes the result of the controlled power-supply. The SCLIB functions are used at the message interchange between two processes.

An example of MGLIB function calls is also shown in figure 3. Here, an initializing function (*sc_open* for the SCLIB) is not necessary, since initialization is carried out automatically at the first call of the MGLIB functions.

B. Source portability

Some problems concerning source portability have



```
main()  /* application process */
{
 int mg(), mg_errend(); /* mglib functions */
 extern int mg_errno;  /* error code holder of mglib */

 rtn = mg( "CUR", "magnet_name", &curr );  /* get magnet current  */
  if( mg_errno < 0 ) mg_errend();   /* error message & exit */

}
```

Figure 3
Example of an application process which controls a magnet power-supply of the KEK linac. Inter-process communications between an application process (workstation 1) and a daemon process (workstation 2) are carried out with the SCLIB functions (see text). Workstation 2 serves as a gateway between the Ethernet and the present control system.

arisen during the development of the MGLIB. The most serious one is the difference of byte order in the expression of numerical values. We have described byte-conversion routines for each computer system with C preprocessor statements.

Since the conversion routines are described in the MGLIB library, application sources are expected to show high portability. Actually, sample programs for basic operation of the magnet power-supplies show perfect portability among the computer systems in table 1. In addition, the basic availability of the SCLIB/MGLIB functions is also checked in a Hewlett Packard workstation with HP-UX, and a Force 68030-based VME system with OS9.

C. Discussion

The typical throughput of a single control action is obtained as 100-400 ms. This value is understood to be the round-trip response time in the present control system.

After introducing the next control system, we should re-develop a new daemon (server) process with a VME environment. It would not be easy since the development of such a server process requires deep understanding of the network parameters and workstations, even with the present SCLIB functions. However, the sources of applications developed so far with the MGLIB functions will be available without any modification, even in the next control system.

IV. DISCUSSION

One of the problems concerning an Ethernet is response delays due to network packet collisions, which inevitably occur in an Ethernet when the network traffic rate is considerably high. An easy answer to this is to replace an Ethernet with a FDDI (Fiber Distributed Data Interface) fiber-optic network system. It is a token-ring type network and is more suitable for a real-time purpose. In addition, we would expect one order better communication throughput since it is capable of a communication rate of 100 Mbit/s. Moreover, it is worth noting that a replacement is possible without any software modification.

The present libraries provide a possibility to use an Ethernet as a high-speed data-transfer network with low cost. There exists a plan to use these libraries for the control system of the TENKO-100, a 100-meter long laser interferometer aimed for detecting gravitational waves now under construction in ISAS (The Institute of Space and Astronautical Science) [8,9].

V. ACKNOWLEDGMENTS

We would like to thank Professor A. Asami, head of the Injector-Linac division in KEK, for his support and kind encouragement. We also thank the staff and operators of the linac for their valuable discussions and assistance.

VI. REFERENCES

[1] K. Nakahara, I. Abe, R.P. Bissonnette, A. Enomoto, Y. Otake, T. Urano and J. Tanaka, Nucl. Instr. Meth. A251(1986)327

[2] K. Nakahara, I. Abe, A. Enomoto, Y. Otake and T. Urano, Nucl. Instr. Meth. A247(1986)153

[3] K. Furukawa, N. Kamikubota, K. Nakahara and I. Abe, proceedings (Int. Conf. on Accelerator and Large Experimental Control Systems, Vancouver Canada, October 1989), Nucl. Instr. Meth. A293(1990)16

[4] K. Furukawa, N. Kamikubota, K. Nakahara and I. Abe, "Upgrade Plan for the Control System of the KEK e-/e+ Linac", this conference

[5] K. Nakahara, I. Abe, K. Furukawa and N. Kamikubota, proceedings (Int. Conf. on Accelerator and Large Experimental Control Systems, Vancouver Canada, October 1989), Nucl. Instr. Meth. A293(1990)446

[6] I. Abe, M. Kitamura and N. Nakahara, "A Diagnostic Expert System in PF Linac Control", this conference

[7] K. Furukawa, N. Kamikubota, T. Shidara, K. Nakahara and G. Horikoshi, proceedings (14th. Int. Conf. on High Energy Accelerators, Tsukuba Japan, August 1989), Particle Accelerators 29(1990)245

[8] N. Kamikubota, K. Furukawa, K. Nakahara, Y. Ogawa and A. Asami, "A study of the Control System for Long-Baseline Laser Interferometer", the proceedings of the Sixth Marcel Grossmann Meeting on General Relativity (MG6) in press, Kyoto Japan, June 1991; KEK preprint 91-124

[9] N. Kawashima, T. Yokota, Y. Ogawa and N. Kamikubota, "Development of 100 m Laser Interferometer Gravitational Antenna (TENKO-100)", the proceedings of the Sixth Marcel Grossmann Meeting on General Relativity (MG6) in press, Kyoto Japan, June 1991

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC09

A Program Development Tool for KEK VME-MAP Control System

H.Nakagawa, T.Katoh, E.Kadokura, A.Akiyama,
K.Nigorikawa and K.Ishii
KEK National Laboratory for High Energy Physics

Abstract

The control system for KEK 12 GeV Proton Synchrotron has been replaced with a distributed VME-bus based microcomputer system and a MAP local area network. In order to simplify programming for network application tasks, a set of a preprocessor for a PASCAL compiler and a network communication server has been developed.

Application programs for accelerator control system have blocks with similar codes; sending, waiting for, receiving, analyzing messages, etc. The preprocessor called "OBJP" incorporates such common codes into the source code written by an application programmer.

In case of a simple program, the size of the source code is reduced by one tenth of a full coding.

I Introduction

The present control system for the KEK 12GeV PS has been modified by using VME-bus based computers and MAP local area network. On these computers, application tasks work under the VERSAdos; real-time multi-tasking operating systems. In this case, tasks in one group run on some computers and they communicate with each other by network. Then, most important factor of such programs is that any programmer can write the communication function of applications easily. By reason of these thoughts, the network support programming tool 'OBJP' has been created.

The 'OBJP' is preprocessor of PASCAL compiler, but the source file of OBJP programming seems to be a new language system like PASCAL.[1] And it also seems like the object oriented programming, but 'OBJP' is not complete object oriented programming. So the 'OBJP' is a communicatable multi-tasking program development tool. But we think that both the 'OBJP' applications and object oriented programming found on a same basic idea; each function is isolated and communicate with each other by message.

Let's show the 'OBJP' programming, and the configuration of the VME-MAP computer system for the 12GeV PS control.

II System configuration

This control system consists of two major devices; 26 VME-bus computers and a MAP local area network.[2] All the VME-computers are linked together in the same level. Their physical connection is bus style, but logical connection is ring style by token-passing. All computers are equal to each other in the logical ring.

Each computer dedicates to each specialized function. There are computers of one network manager, one program development, four console and many device controllers as shown in fig.1.

The network manager is "UNIX", which checks computer condition and makes logging of its information. The computer "ROLA" offers multi-user programming environment, and sends applications to other computers at rise-up of the computer. The console computer works for human interface. The device controllers control the power-supplies and monitor their status.

Video information signals are also transmitted through the MAP network cable.

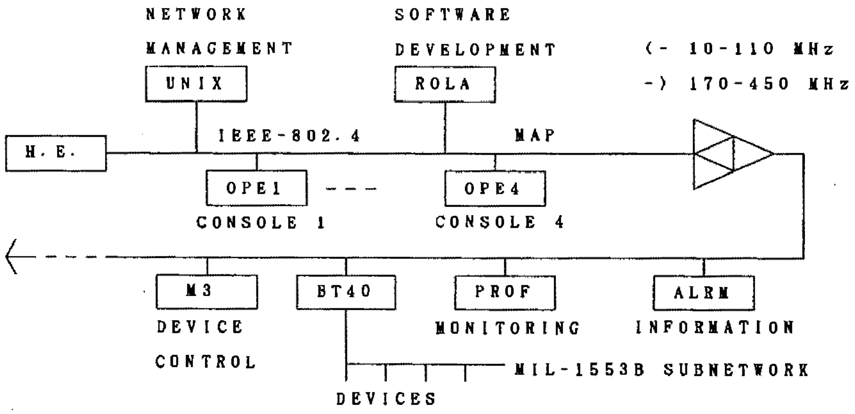


Fig.1 A scheme of MAP-VME control system.
The computers are connected with the MAP local area network.
The head end remodulator(H.E.) repeats signaling from the reverse channel on the forward channel.
Each box shows a computer and ID named after the function
There are some broad-band amplifiers.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC09

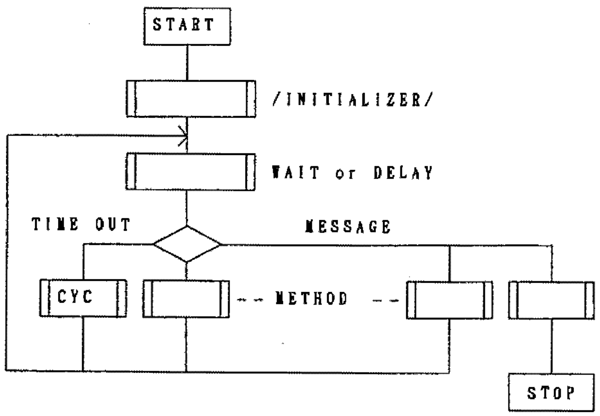


Fig.2 Typical flow chart of resident tasks for control. Similar codes for initialize part, wait for events and event select appear in many programs.

III "OBJP" programming

The OBJP preprocessor program has been created as a result of a research of program design of multi-task system. We notice the appearance of similar codes in many programs.

Fig.2 shows that a flow chart of a program which runs on multi-task OS and residents on memories. In the chart, both "WAIT or DELAY" part and message select part are required from all our tasks. And almost all of our tasks must send messages.

Our programmers are not proper in programming, then good programming environment is required. So, we avoid the method by editor, so that the method by means of automatical formation has been chosen. For this purpose, the preprocessor called "OBJP" has been developed. The OBJP makes a PASCAL source as shown in fig.3.

Simple program of OBJP source is shown in fig.4, . It works really in our system for relay-board control. This program is simpler than one in which all possible items relating to the network communication functions are written.

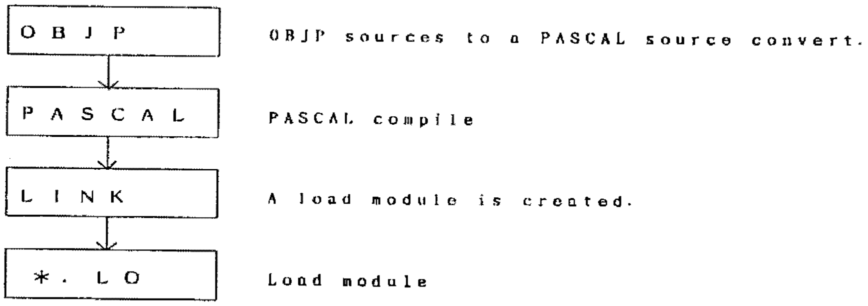


Fig.3 Program development flow.
The OBJP is a preprocessor of PASCAL compiler.

```
SWITCH = W
   ( Graphics and Delayw      )

program SOUNDPRO ( Objective pascal )  ;

/COMMON/
const
TASK    = 'WAAA'        ;
TDELAY = 000            ; ( msec )

/LOCAL/
const
   DDTT = 500    ; ( msec )
   DMADA= 0500   ; ( msec )
   DELAY = 21    ; ( Delay for RMS )

type
   MSDATA = string[ 255 ] ;

var
   ADDR [ origin 16#DFF502 ] : word ;
   ADDS [ origin 16#DFF500 ] : word ;
   I : integer        ;
   J : integer        ;
   K : integer        ;
   L : integer        ;
   N : integer        ;
   ERROR : integer    ;
   MM : MSDATA        ;
   S_NAME : CCCC      ;

/EXTERNAL/
0.OBJP.MTOBIN.TX

/INITIALIZER/
L := 16#DFF400         ;
N := 16#400            ;
S_NAME := 'ROUT'       ;
```

```
ERROR := MakeMMIO( S_NAME , L , N )   ;

/METHOD/
TERM:
        1:DCLK <:= 'MESS TERM WAAA' ;

OHHI:
        MM := MESSAGE          ;
        I  := MTOBIN ( MM ) ;
        J  := MTOBIN ( MM ) ;
        K  := 1                ;
        if I <> 0 then
           for N := 1 to I do
              K := K * 2       ;

        for L := 1 to J do
          begin
          ADDR  := K           ;    ( relay on )
          ERROR := TRAP1X(DELAY,DDTT);(wait )
          ADDR  := 0           ;    ( relay off)
          ERROR := TRAP1X(DELAY,DMADA);(wait)
          end                  ;

REST:
        MM := MESSAGE          ;
        I  := MTOBIN ( MM ) ;
        K  := 1                ;
        if I <> 0 then
           for N := 1 to I do
              K := K * 2       ;

       ADDS  := K           ;    ( relay on )
       ERROR := TRAP1X(DELAY,DDTT);(wait )
       ADDS  := 0           ;    ( relay off)
       ERROR := TRAP1X(DELAY,DMADA);(wait)

/END/
```

Fig.4 Sample of a OBJP source is shown.
A program for driving a relay-board is shown as simple example.

IV Servers

The simple description about the OBJP applications is supported by communication server tasks. They work on all computers and communicate with each other. Simplified diagram is shown in fig.5.

The task '&SRV' receives the request for message send, and it passes the message to task 'C37X' which controls communication board. The task 'NANN' is an agency for system call on remote node. Then a message from an application is placed in a receive task's communication buffer. There are more servers for network management or file manipulation.

At OBJP level programming, any programmers do not have to know the server's functions.

V Standard human interface

Touch screens and CRT displays are used for the human interface of the accelerator control. The touch screen system has been managed by a special task called 'GPAN'. Application programmer need not write a part of program for controlling the touch screen.

All commands for the control has been written in files for the GPAN. When the surface of screens is touched, the GPAN sends a message to a task of either device control or information display. Scheme of this relation is shown in fig.6. Data structure of the input file is standardized as shown in fig.7.

When the GPAN is used, programmers need not write command request function of programs. That simplifies programming.

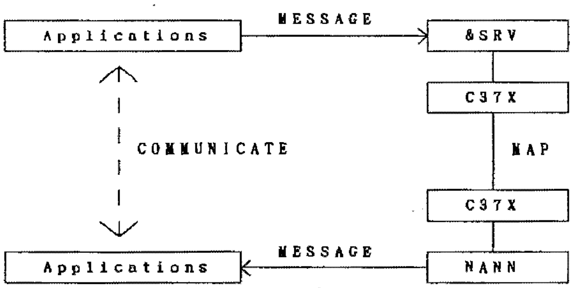


Fig.5 The communication servers. Applications which is made by the OBJP communicate with each other through communication servers.
When MAP protocol requires connections, the applications do not have connections. The server only has ability of the connection.

VI Typical application task

When the GPAN sends a message to a task, the message brings about cascade shower of messages among some tasks. A sample is shown in fig.8, where many tasks work for controlling a beam slow extraction of both EP1 line and EP2 line. Each task has only one function, so that it is very simple.

In this case, the one task corresponds to one device similar to an object of the object oriented programming. Because one device replacement corresponds to one task replacement, good program maintainability can be kept.

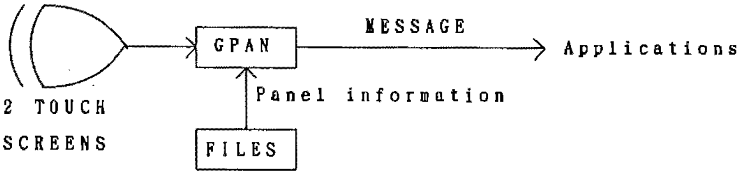


Fig.6 Touch screen management is done by special task, called 'GPAN'. The 'GPAN' reads "Control Information Files" and rearranges buttons of the touch screen.
If the button displayed on CRT is touched, the 'GPAN' sends a message to the target task.

```
type

button = record
 number  : word;          { Button location XY  }
 corrv   : word;          { Button change style }
 ccodep  : word;          { Color code : normal }
 ccoder  : word;          { Color code : pushed }
 infop   : string[27];    { Letter : normal     }
 infor   : string[27];    { Letter : pushed     }
 taskname: string[8];     { Receive task name   }
 commmess: string[30];    { Message 1           }
 tasknamr: string[8];     { Receive task name   }
 commmesr: string[30];    { Message 2           }
 keykey  : word;          { Keybord or 2nd panel}
 nextssf : string[8];     { Next screen         }
end;
```

Fig.7 A control data structure for our standard touch screen.
Data size per a touch button is about 200 bytes. The data are edited by a program for the exclusive use of data making.

The sample is complex case. One program is send to two computers for both EP1 and EP2. But in many case, a pair of tasks works one job; one task controls devices or takes data and another task displays the status on a CRT display. The structure depends on a programmer's conception.

VII Sample of remaking

As hardwares of accelerators are replaced often, control program for a replaced hardware must be also replaced. For example a sound information unit has been replaced recently, as it is hopeful that useful messages for operators are announced. The step of the replacement is shown in fig.9.

In this case, there are 3 stages. First, a relay-board on VME-bus drives talker units which can talk only one message of 8 seconds. It is very simple, but not flexible.

We introduce a new alarm message system. We have made the talker of both a D/A board and its drive program. As the source program of OBJP is simple and many functions does not appear in the source codes, the simple modification has been made.

The third stage in fig.9 is on testing now. It is a logic table based information system. We think that it is similar to knowledge base system. This program calculates input states relation and it sends a message when the condition becomes equal states written in its table. In this case, the new talker system can use without modification when numbers of information are increased.

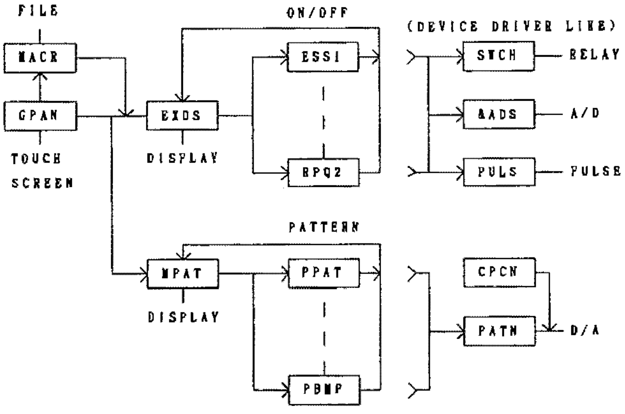


Fig.8 The OBJP applications work communicating with many tasks. This is a sample of beam extraction control system. Each box means a task with a special duty. These tasks work on some computers.

VIII Conclusion

In order to make up the new system by a few persons , a simple programming method is required to make many programs. We have made the OBJP for simple programming as the result of research of multi-tasking system.

In case of making program with "small and many" tasks, the one task corresponds to one device, so that it is similar to an object of the object oriented programming. But this programming method is not so easy for many programmers. They makes complex programs which have all functions in one task. "The all in one type programming" seems to be high performance, but debugging process is very complex. The "simple and many" program's debugging is simple, because the function of a task is simple. Total performance of the "simple and many style programming" becomes higher than the "all in one type programming", because of the simple debugging and its portability to other jobs.

IX Acknowledgement

The authors wish to acknowledge the continual and helpful cooperation of the 12GeV PS members. We wish to thank Dr. T.Kawakubo for his useful programs and his suggestions for debugging of the OBJP .

X References

[1] H.Nakagawa, "OBJP," KEK Report 90-20, Feb. 1991, (in Japanese).

[2] T.Katoh, H.Nakagawa, K.Ishii and E.Kadokura, "VME-COMPUTER BASED CONTROL SYSTEM FOR THE KEK PROTON SYNCHROTRON," Europhysics Conference on Control Systems for Experimental Physics, Villas-sur-Ollon, Switzerland, 1987, and KEK Preprint 87-91.

```
@ 1ST STAGE

INPUT --> RELAY DRIVE PROGRAM --> RELAY --> BUZZER OR
                                            FIXED TALK UNIT
                      |
                      |  A FEW MINUTES
@ 2ND STAGE           |
                      V
INPUT --> BIT TO TERMS TRANSLATOR --> TALK PROGRAM --> D/A
                                           |
                                           |  SAME PROGRAM
@ 3RD STAGE                                |
                                           V
INPUT --> INFORMATION BY LOGIC TABLE --> TALK PROGRAM --> D/A
                       |
                       |--> INFORMATION DISPLAY --> DISPLAY
                       |
                        --> LOGGING PROGRAM --> PAPER OR FILE
```

Fig.9 A sample of task replacement.
A new talk information task for alarm announcements is introduced in the control system.
Stage 1 : fixed talker units were used for sound information.
Stage 2 : a new talker has been introduced into the system.
The relay drive program has been modified for new system.
Stage 3 : a new information system(now testing).
The program is like a core of any knowledge base system.
The new talk task can be used without modification.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC10

Network Performance for Graphical Control Systems

Peter Clout, Mark Geib, and Robert Westervelt
Vista Control Systems, Inc.
134B Eastgate Drive, Los Alamos, NM 87544, USA

Abstract

Vsystem is a toolbox for building graphically–based control systems. The real–time database component, Vaccess, includes all the networking support necessary to build multi–computer control systems. Vaccess has two modes of database access, synchronous and asynchronous. Vdraw is another component of Vsystem that allows developers and users to develop control screens and windows by drawing rather than programming. Based on X–windows, Vsystem provides the possibility of running Vdraw either on the workstation with the graphics or on the computer with the database. We have made some measurements on the cpu loading, elapsed time and the network loading to give some guidance in system configuration performance. It will be seen that asynchronous network access gives large performance increases and that the network database change notification protocol can be either more or less efficient than the X–window network protocol, depending on the graphical representation of the data.

I. INTRODUCTION

Performance is one of the considerations when configuring computer control systems. Other considerations are equipment and software costs. In order to help our customers to make intelligent decisions we have made some initial performance measurements on network real–time database access for both synchronous and asynchronous remote access, as well as some measurements to compare network database change notification against network X–protocol for graphical data presentation.

II. VSYSTEM'S REAL–TIME, NETWORKED DATABASE

Vaccess is the real–time database component of Vsystem [1]. A library of access routines allows for full access to the run–time database. Routines are included to search the database in various ways and to request and cancel change notification by wake–up or interrupt routine (AST) execution. The library of access routines handles the network transparently to the user. Network access can be either synchronous or asynchronous.

With synchronous access to a remote database, the program making the Vaccess routine call will not continue execution until the request has been sent over the network and a reply received, a process that can take many milliseconds. The network messages contain few useful bytes leading to inefficient cpu and network utilization.

With asynchronous access, a program can make many calls and then call a wait routine, at which point the program will not continue execution until all the calls have been completed. Not requiring an immediate answer means that the Vaccess routines can include many, if not all, the remote database access requests in a single network packet with a consequent dramatic increase in cpu and network utilization.

The arrival of X–windows and X–terminals has recently given the implementers of graphical control systems more configuration options. One can use workstations all running the graphics software and all accessing the data over the network, or one can use a single, powerful, processor running a single copy of the graphics software and serving the users at X–terminals. Both configurations have advantages and disadvantages. Here we attempt to quantify the network issues in this choice.

III. REMOTE DATABASE ACCESS MEASUREMENTS

Vsystem supports VAX/VMS and VAX/ELN. All of these measurements were made between two VAXstation 3100 model 30 workstations rated at 2.7 VUPs, running VAX/VMS V5.4–2, current VAXstations have performances about four to five times that of the VAXstation 3100 model 30. It is important to note that these measurements were intended to compare the different protocols and they can never replace benchmarks on the proposed computers as many factors affect the system performance apart from the cpu rating.

The network protocol used by Vsystem in these measurements was DECnet. The test consisted of a program running in one VAXstation 3100 model 30 which called multiple "RPUT"'s and "RGET"'s to a database channel in a database on another VAXstation 3100 model 30. "RPUT" and "RGET" are Vaccess library calls to put and get a real number to and from a channel. Standard VMS library calls were used to access the elapsed time and cpu time, and VMS NCP was used to report the network bytes and messages.

Table 1 on the following page lists the measurements made for 1000 calls. Measurements for larger and smaller numbers of calls scaled with the measurements in Table 1.

	Kbytes Sent	Kbytes Received	Messages Sent	Messages Received	CPU Sec	Elapsed Sec	Kbytes/ CPU Sec	Kbytes/ Elapsed Sec
Synchronous RPUT	28	16	1,000	1,000	3.73	8.88	7.5	3.2
Asynchronous RPUT	28	0	21	0	0.54	0.57	52	49
Synchronous RGET	20	24	1000	1000	4.31	9.31	10.2	4.7
Asynchronous RGET	20	24	15	17	0.87	1.14	50.6	38.6

NOTE: All measurements for 1000 calls

Table 1: Performance Measurements for Network Database Access

The column showing the number of messages sent and received clearly shows the effect of asynchronous calls, and the effect on cpu overhead can also be seen in the cpu column.

Figure 1 summarizes the results in terms of the throughput against the demand in requests/second. The asynchronous RPUT performance is improved by the lack of need for replies and that accounts for the high throughput. Asynchronous RGET calls return the value and hence the approximately halved performance.

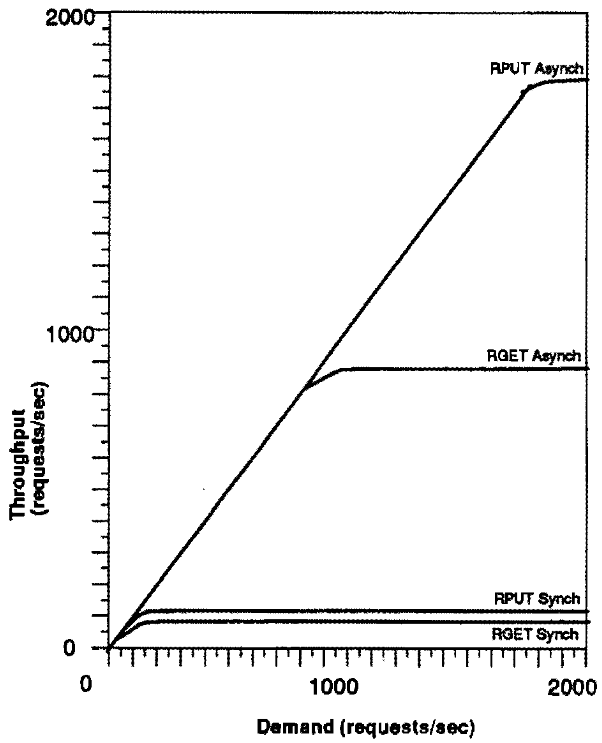


Figure 1: Remote Database Access Performance Between Two VAXstation 3100 Model 30 Computers

The remarkable impact of the requirement of synchronous calls to wait for a reply is shown in Figure 1. The network overhead associated with sending each request as a separate message is about six times the basic call overhead. The elapsed time is further increased over the increase in cpu time by the need to wait for a reply on each call.

IV. NETWORK LOADING

If one considers the message overhead of 48 bytes in addition to the 20 bytes in the "RGET" request message and the 24 bytes in the reply, a total of 140 bytes on the network is associated with each "RGET". Allowing an arbitrary 50% network loading and assuming no collisions, one can achieve a total network throughput of about 4,500 "RGET"'s/second. With asynchronous "RGET"'s this network throughput can be doubled as about 50 "RGET" calls can be included in one network message with the 48–byte overhead. Using the same arguments, asynchronous "RPUT"'s will have the bandwidth further doubled to about 18,000 "RPUT"'s/second because there is no reply message.

All the Vsystem tools use a feature in Vaccess by which any process on the network can request notification of a significant change in any field in a channel. Thus, Vdraw will take a local copy of the value and request notification of a significant change. It is then the responsibility of the remote database to notify Vdraw of the change when it happens. This ensures that the network and computer processing bandwidth is only taken with required information. With change notification, the need for polling is removed and while the network throughput will be about 4,500 changes/second (the network loading of a change notification is about twice that of an asynchronous "RGET") those changes will all be significant, requested changes rather than mostly checks on unchanged data.

V. GRAPHICS MEASUREMENTS

For these measurements the same VAXstation 3100 model 30 computers were used with one containing a Vaccess database and a program to generate changes in that database. For the Vaccess communications measurements, Vdraw was run in the other VAXstation 3100 model 30 and the network traffic measured as the data was displayed with different Vdraw control tools. The tools used were a text display, a bar displaying two values, the second value as a small triangle, and a slider with a readback channel. For the X–window network protocol measurements the same test was run but with Vdraw running on the same VAXstation 3100 model 30 as the database and the other VAXstation 3100 model 30 acting as an X–server. In this case, the network traffic was the X–protocol. In this case measurements were made with the X flush rate set both fast, 0.01 seconds, and slow, 10 seconds. Different flush rates did not change the number of bytes transferred but dramatically changed the number of messages. When the flush rate was set slow, the network messages were full. Table 2 shows the results of these measurements and compares the number of bytes transferred.

Text Display

	Bytes/ Update	Updates/ Message Sent	Updates/ Message Received
Vaccess	85	16	39
X–Protocol Slow Flush Rate	48	28	103
X–Protocol Fast Flush Rate	48	1.7	10

Bars

	Bytes/ Update	Updates/ Message Sent	Updates/ Message Received
Vaccess	168	8.2	28
X–Protocol Slow Flush Rate	81	16	26
X–Protocol Fast Flush Rate	81	11	13

Sliders

	Bytes/ Update	Updates/ Message Sent	Updates/ Message Received
Vaccess	168	8.2	28
X–Protocol Slow Flush Rate	375	3	3.5
X–Protocol Fast Flush Rate	375	0.2	0.5

Table 2: Network Performance for Vaccess and X–Protocol Communications

For simple graphic objects, the X–protocol is more efficient in network usage than the Vaccess AST protocol. One reason for this is the amount of data transmitted with the AST to the requesting process in addition to the changed value. This is to minimize the possibility that the process will have to access the remote database further in order to complete the processing of the change. The difference in network efficiency also reflects the optimization in the updating of text and slider data displays [1]. Bar updates can get considerably more complex depending on the options chosen. The large number of bytes transferred to update the slider reflects the complexity of the slider and the fact that it displays the data in graphic and text form. Here the Vaccess mechanism is more efficient in terms of network loading.

Vdraw offers the user the possibility of displaying information with objects of arbitrary complexity and the network loading could become a factor in designing and configuring a system.

VI. CONCLUSION

From the measurements it is quite clear that wherever possible, asynchronous database access should be used. When data is being monitored over the network, the performance increase in using change notification will of course depend on the number of changes but in most systems the increase in performance will be dramatic.

The measurements of the network loading of the X–protocol are quite striking in their efficiency. The setting of the X flush rate was observed to affect the perceived performance of the graphics.

Further measurements will be made to understand system performance and identify areas for optimization.

VII. REFERENCES

[1] The Influence of Industrial Applications on a Control System Toolbox, Clout, P. *These Proceedings*

A New Approach in Development of Data Flow Control and Investigation System for Computer Networks

I.Frolov, A.Vaguine, A.Silin
USSR Academy of Sciences
Moscow Radiotechnical Institute
Warshawskoe shosse, 132, 113519 Moscow

ABSTRACT

This paper describes a new approach in development of data flow control and investigation system for computer networks. This approach was developed and applied in the Moscow Radiotechnical Institute for control and investigations of Institute computer network. It allowed us to solve our network current problems successfully. Description of our approach is represented below along with the most interesting results of our work.

INTRODUCTION

Seven years ago we started the development of a new control system for an experimental electron accelerator in our institute. It was planned at the beginning to apply six computers PDP 11/70 and six computers PDP 11/23. These computers were interconnected by lines DL KI/SI (These lines were developed and manufactured in the USSR. The throughput is about 500 kbit/sec). The operating systems are RSX11M/S.

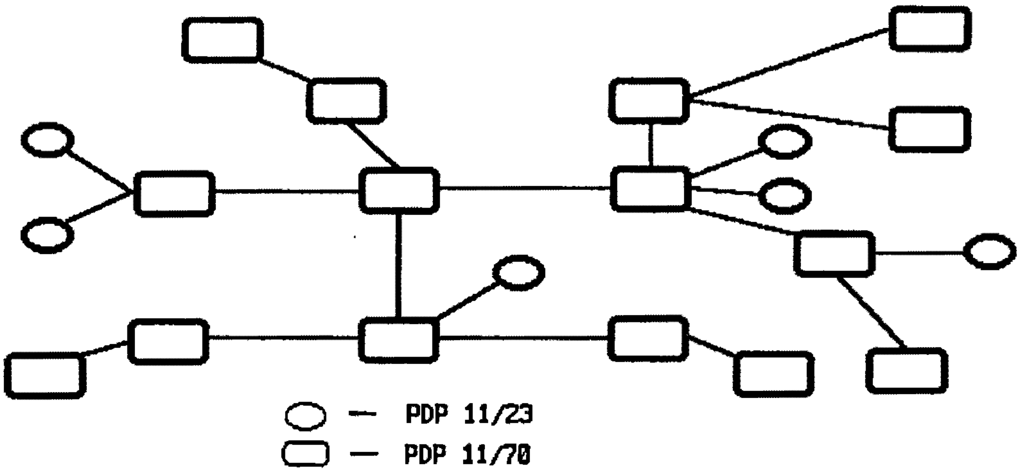


Figure 1. Network structure (1988).

The main our problem was a choice of network software. In that time we knew the DECNET and the so called ALISA, and had two class of software simultaneously on each computer. One week we used DECNET and next week ALISA. The DECNET is a more powerful network package, but the main problem for us was a limited size of available operating memory. First of all it is valid for PDP 11/23. The compromise was as follows: ALISA as a first step, and DECNET for our perspective, when we will get a more powerful computers.

OUR PROBLEMS

It was planned to control the accelerator from a dedicated console. Console consists of four graphical stations. Each graphical station is a ordinary network node (from viewpoint of network service). In addition one computer was planned for a database management. And the first task for us was to investigate application efficiency of network graphical stations and network database. At that time first users of our network appeared. This period is characterized as a period of development and debugging of software for accelerator automation. Terminals and computers were placed in different rooms. And often the main problem for users was to find a free terminal. If free terminal was connected to another computer he used either virtual (remote) terminal service or virtual disk. Also network services were used for transfer of files, and copying of disks and magnetic tapes. In addition another problem for us was the development and enhancement of our network. Many colleagues from other experimental systems asked us to connect their computers to our network for use of graphical stations, plotters, densitometer. Figure 1 illustrates a structure of our computer network (1988). And the vital problem for us was the creation tools for network investigations to solve correctly all our problems.

PROBLEM of INVESTIGATION

First of all the problem of network study for us was the problem of network traffic research. A conventional approach for the message flow description is based on data flow representation in terms of a distribution function of time intervals between the two adjacent messages . The advantage of the approach lies in the use of the queueing techniques. However , the utilization of the approach in real networks traffic representation seems to meet with difficulties. Thus another approach was used in message flow description for communication networks . A pair of counters is associated with a network node . One of them is incriminated in the case of a message reception in the node whilst the second is incriminated in the case of a message delivery . Both receiving and transmitting parts are equipped with counters which have to measure the lengths of messages . Within preset time intervals (e.g. every second) the counters are sampled . The differences between current and preceding readings feature numbers of transmitted and received messages (i.e. the speed) .

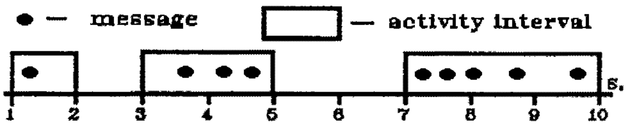


Figure 2.

The set of adjacent unit intervals with zero number of transmitted (received) messages is considered as inactivity interval . On the contrary , the set of adjacent unit intervals with number of transmitted (received) messages more null is called the activity interval. Figure 2 illustrates our new definitions. Spaces between activity intervals are the intervals of inactivity.

Respectively , the communication network flows are described by distribution functions of activity intervals , inactivity intervals , message reception (transmission) rates and lengths of messages .

The structure of network software for one networks node is shown on Figure 3. System receive information about messages from standard QIO requests. Standard QIO request issued from user task are directed by operational system to device driver. The ALISA device drivers were edited to contain counters and a program was developed to sample information from this counters every second and to modify distribution functions. In addition a program was developed for demonstration of all current traffic information on user terminal. The size of the program to sample information is about 5 kbyte. The influence of this changes on network throughput are less than one percent.

RESULTS of INVESTIGATIONS

If we take into account network software and hardware delays, the network rate between two adjacent nodes is about 15 kbyte/sec for messages of 512 byte length. It's enough for normal work in the regimes of virtual terminal or virtual disk. In this case user don't see distinction from situation when terminals or disks are connected directly to computer. Below most interesting results of investigation will be represented.

A. Virtual terminal service

We investigated about 30 users. Users did not knew about our researches Figure 4 illustrates the histograms of inactivity and activity intervals and transmissions rates histogram obtained from one user. The user activity was a development and debugging of programs (edition process). Investigations showed that such histograms are similar for all users who have a little experience with computer keyboard and editor. The average inactivity intervals for this category of users are between 8 - 12

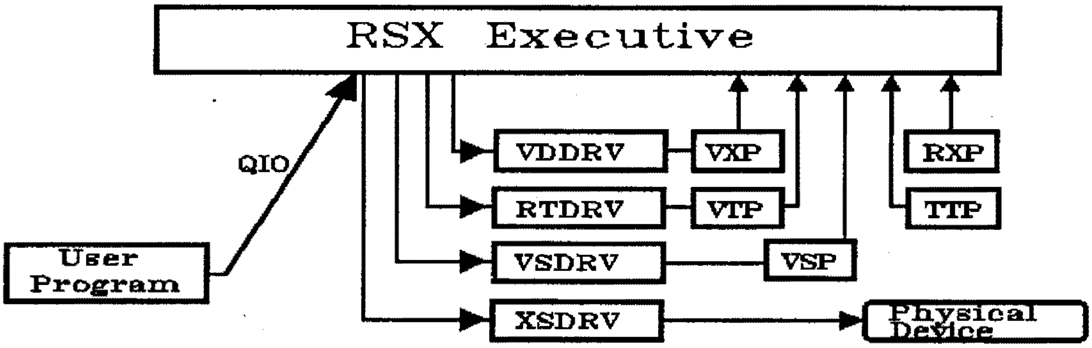


Figure 3. RSX ALISA Structure.
VXP <=> RXP - Virtual System Device Processes
VTP <=> TTP - Virtual Terminal Processes
VSP - Network Communications Process

seconds. For unskilled users this intervals are about 20 - 30 seconds. Figure 5 shows the same histograms obtained from two users. They used the same line simultaneously.

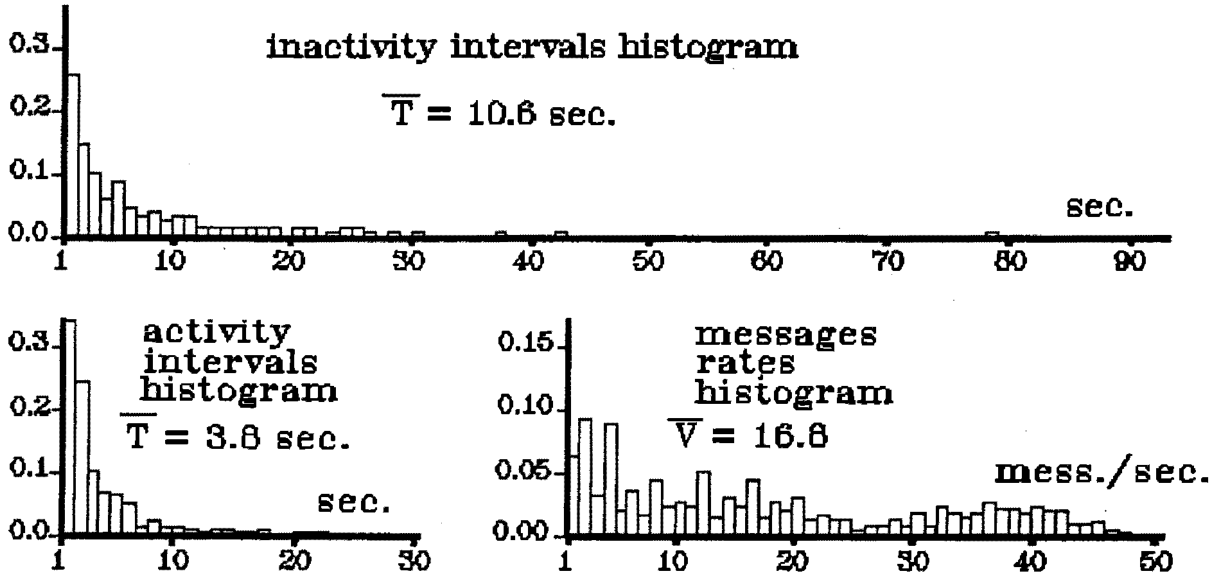


Figure 4. Virtual terminal service (one user).

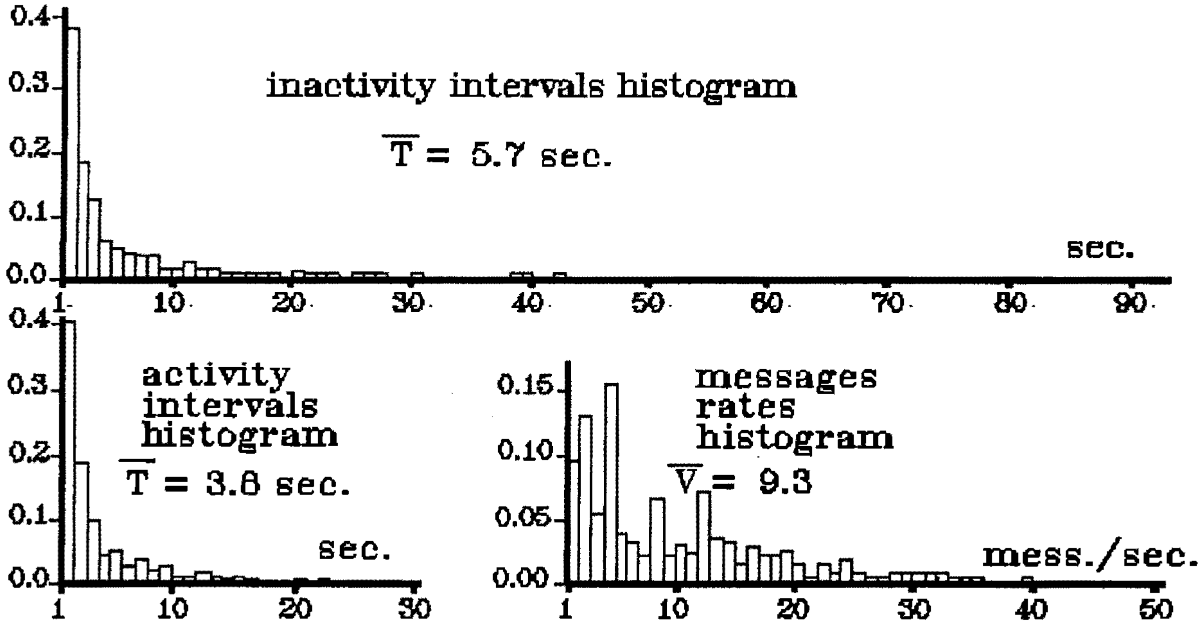


Figure 5. Virtual terminal service (two users).

As we see the average value of inactivity time interval reduced practically by a factor of 2 but the average value of activity interval essentially doesn't changed. This result can be easy explained from Figure 4. The proportion between average values of inactivity and activity intervals are about 3/1 - 4/1. Figure 4.1 illustrates distribution function for lengths of messages in the virtual terminal service. It' s difficult to represent this result as a histogram. We have investigated many cases and we can say that these digits are approximately the same in all cases.

Lengths	<50	<100	<150	<200	<250	<300	<350	<400
Share	0.89	0.088	0.004	0.001	0.003	0.01	0.0	0.0

Figure 4.1.Distribution Function for Lengths of Messages.

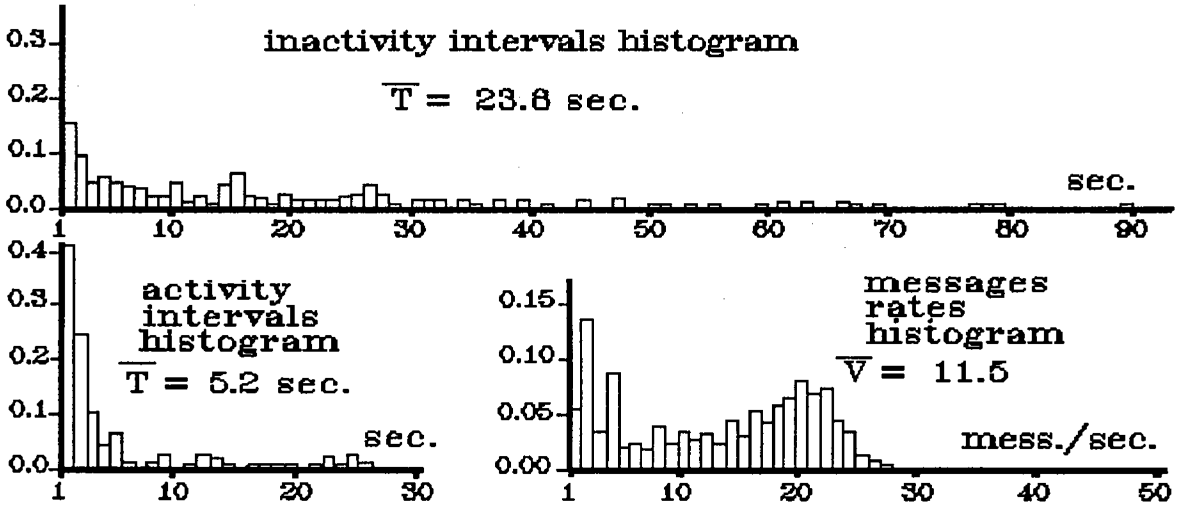


Figure 7. Virtual disk service (one user).

B. Virtual disk service

Figure 7 shows the set of histograms obtained when one user used virtual disk service for programs development and debugging. The average time of inactivity intervals is about twice that for virtual terminal case. The situation for two users is similar to the situation with virtual terminal service. Figure 6 shows the transmission rate histogram for case of copying one disk to another through the network. We can't account for this result but we think it will be of interest for other colleagues.

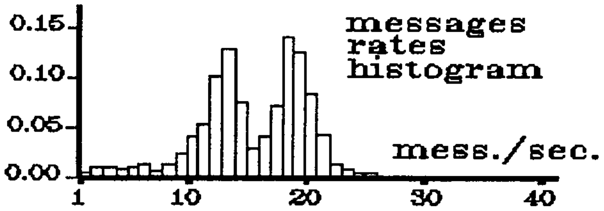


Figure 6. Disk Copy through Network.

C. Virtual graphical station

We investigated many cases of experimental data processing on graphical station. The processing is a visualization of two dimensional matrix (256 X 256, 64 X 64 and others), graphics drawing, search and generation of cross sections and so on. It's interesting that the obtained results are analogous to the results for virtual terminal investigations. In all probability to explain this results we need take into account the characteristics of men's mental processes.

CONCLUSIONS

It was very useful for us to have the system for investigations our computer network. We obtained working tool for normal work. We have acquired possibility to make accurate decisions and pleasure to understand it. Any time we need to connect next computer to network we haven't problems. We decide problem on the common sense level but not the level of complex theory. Of course such approach is true only for a little computer networks (a few dozens nodes). Our investigations have shown that for a little networks the flows in lines are far from Puasson type. Our approach can be useful for networks with insufficiently fast lines, when the problem of overloading are vital, but for users with fast communications lines the problem of network investigations don't exist. Meanwhile we hope that our results will be of interest for network experts from academic viewpoint and will be helpful for psychology.

Concurrent Control System for the JAERI Tandem Accelerator

S.Hanashima, T.Shoji, K.Horie and Y.Tsukihashi

Japan Atomic Energy Research Institute
Tokai-mura, Naka-gun, Ibaraki-ken 319-11, Japan

Abstract:

Concurrent processing with a multi-processor system is introduced to the particle accelerator control system region. The control system is a good application in both logical and physical aspects. A renewal plan of the control system for the JAERI tandem accelerator is discussed.

Introduction

Progress in the micro electronics region makes real concurrent(parallel) processing reasonable for various applications today. In the logical aspect, a control system of a particle accelerator is a combination of many processes concurrently working for control and monitoring. This implies that the control system may be a good application of the concurrent processing. In the JAERI tandem, we are working to renew the computer control system. A multi-processor system and concurrent programming language will be used in the new system.

Concurrent processing in the control system

We can treat a control system of a particle accelerator as a set of processes to monitor and to control many devices. They have different tasks(roles) weakly coupled with each other. We can depict the system with a model of processes communicating with each other by message transmission, called communicating process architecture(abbreviated as CP architecture)/1/. A concurrent programming language based on the model simplifies programming of the control system, because of easy description of the intrinsic concurrency and communication in the system.

On an usually available multi-tasking operating system, it is possible to describe the above concurrency. But it is not practical because of the overhead to manage too many processes and communications. Thus we must divide the processes only at a moderate level and convert many concurrent processes to several sequential programs in the actual implementation. The resultant programs are difficult to understand. The multi tasking operating system is not ideal for the above modelling.

The merit of concurrent programming is enhanced by use of multi-processor. The concurrent program distributed on multiple processors can give us dramatical merit of performance, because computing power of the multi-processor is proportional to the number of processing elements. It depends on the concurrency of the application(explicit or implicit) and current state of the computer technology. Digital Integrated circuit technology is advanced to increase density of circuits and the performance. Today, high-performance micro processors are available with a low cost.
Well organized multi-processor system has good cost performance ratio.

Communications are important in the system. They are ones between processors and ones between processor and external elements. Not only high transfer rate of the data, but also short leading time are necessary to get good response.

We are interested in Transputer/2/, OCCAM/3/,/4/ and CP architecture as a base. OCCAM and transputer were developed together by INMOS limited as parallel programming language and micro-processor to execute the concurrent program respectively. Both are commercially available with a program development system.

OCCAM allows description of parallel processes as easy as sequential process descriptions in a usual programing language. Communications between parallel processes are made through message transfer mechanism called channels.

The transputer supports concurrent processing very efficiently. Its latency to switch context is less than 1 micro seconds. And it supports message passing mechanism by hardware. For external communication, it has 4 serial communication links called INMOS links. It is optimized to execute OCCAM program. Transputer and OCCAM are based on CP architecture theory. Transputer executes OCCAM program very efficiently.

In addition to OCCAM language, today other concurrent languages are available for transputer. One of them is Par.C language which is a parallel extension to the C language and has advantage against OCCAM on defining complexed data structures.

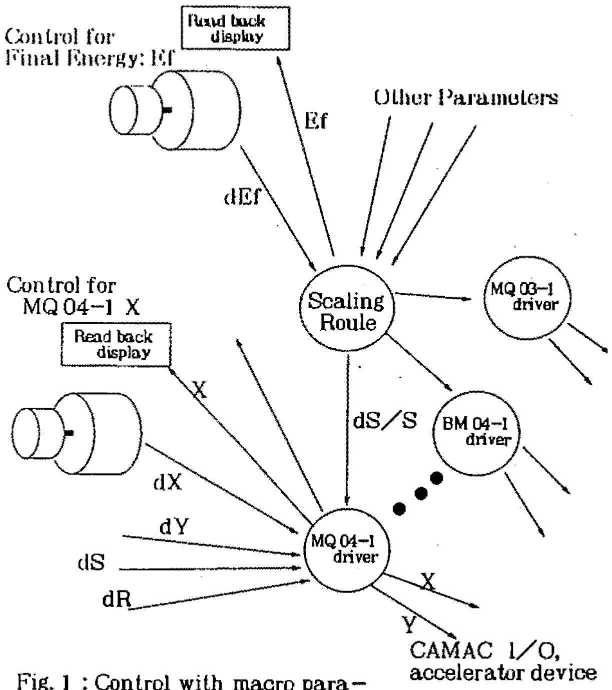


Fig. 1 : Control with macro parameter, Ef(final energy of particle)

Needs for Computing Power

Today, a new particle accelerator is usually controlled by a computer control system. With an appropriate network, it offers low cost remote control from the operator console to the accelerator devices. It also helps the operation with its programmable computational power and mass data storage. High level languages and high level symbolic access methods to the accelerator's data points have been introduced to emphasize the advantages of computer control as mentioned above. It has simplified programming of accelerator for accelerator engineers and physicists, and makes the role of the control system more important than before.

But these benefits have been obtained at the expence of slow response to the operator's action, even in a very primitive manipulation of the devices. In larger accelerator systems, there are more data points and data classes and the network hierarchy is deeper. So the resultant response has become slower. To improve the situation, more computing power and better data throughput with good real time response are needed. With larger computing power, we can add many functions such as flexible feed back control with digital signal processing(DSP), linked control of many data points with a few macroscopic parameters and so on.

One of the good examples is the linked control of data points with a few macroscopic parameters. In the JAERI tandem, we must tune about 50 beam optical parameters, to transport the particle beam to the target. Readjustment is needed for 30 data points of them, when we change final energy of the accelerated particle. Figure 1 shows process diagram of the linked control with macroscopic parameter, the final energy of particle(Ef). A scaling rule process accepts macroscopic parameters, the final energy(Ef), mass of the particle etc.. It calculates particle energy at each optical device position and adapts appropriate scaling rule/5/ and sends the increment(or decrement) signal to each device driver process. The drivers output low level control signals to the actual accelerator devices. When operator turns the valuator of the final energy, all beam optical parameters follow it and the beam current on target is maintained.

For comfortable real time response, the valuator value must be sampled and processed 10 to 20 times per second(or

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S09DPP01

faster). Concurrent processing with multi-processor achieves it, distributing processes to several processors.

Design of the Concurrent Control System for the JAERI Tandem

Figure 2. shows the image sketch of the new system. The control center of the system consists of engineering work stations(EWS's) and a central processor colony. The colony is a multi-microprocessor system. Each processing unit(PU) in the colony is a transputer(T800 INMOS Limited) based processor. It consists of 32bits transputer with 64bit floating point hardware and local memory. They are linked with each other through serial communication links. The colony executes most time critical processings. It is programmed with concurrent programming language.

The EWS's perform several functions. One of them is CRT display for man-machine interface. A window system on the EWS is used for the purpose. The second is data base. A relational database management system is used. The third is a host processor for the central colony.

The front end interfaces to the accelerator devices are CAMAC modules. They will be ones working in the current control system(16 crates are working on 5 serial highways). In the new system, the serial highways will be driven by the central colony through serial high way drivers, which also have transputers as control processors.

Conclusion

The multi-processor concurrent processing is introduced in the particle accelerator control system. With appropriate modelling methodology, the control system is programmed naturally in the multi processor and very high speed processing is obtained. As an example, renewal plan of the control system for the JAERI tandem is discussed.

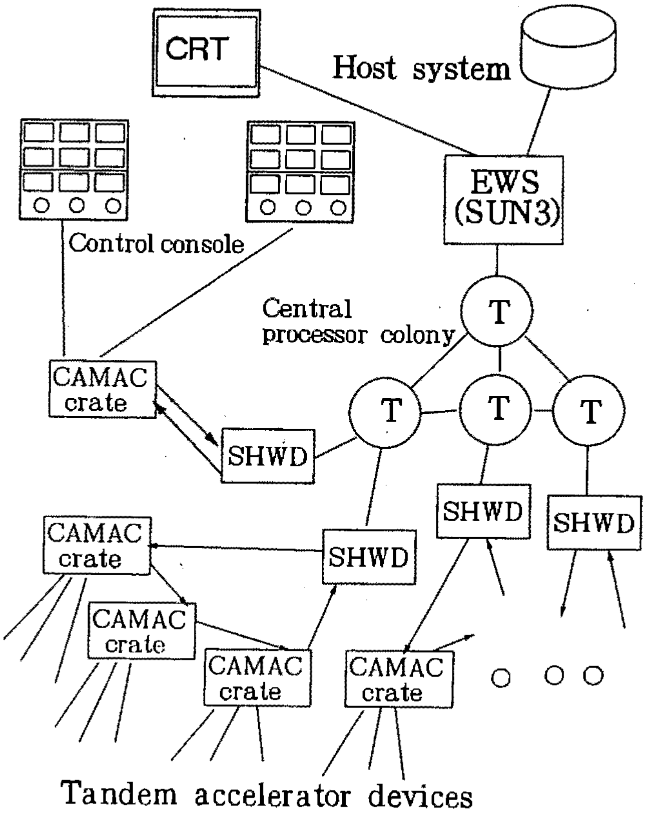


FIg.2 Image sketch of the concurrent control system for the JAERI tandem accelerator

References

1. INMOS limited: Communicating Process architecture, Prentice Hall(1988)
2. INMOS limited: The Transputer Data Book,(1988)
3. INMOS limited: OCCAM2 Reference manual, Prentice Hall(1988)
4. G. Jones and M. Goldsmith: Programming in OCCAM2, Prentice Hall
5. S. Hanashima and E. Minehara: Rev. Sci. Instrum. 57,5(1986)

Palantiri:
A distributed real-time database system for process control.

B.J. Tummers, W.P.J. Heubers
bas@nikhefk.nikhef.nl, wimh@nikhefk.nikhef.nl
National Institute for Nuclear Physics and High-Energy Physics, section K
P.O.Box 41882, 1009 DB Amsterdam, The Netherlands

Abstract

The medium-energy accelerator MEA, located in Amsterdam, is controlled by a heterogeneous computer network. A large real-time database contains the parameters involved in the control of the accelerator and the experiments. This database system was implemented about ten years ago and has since been extended several times. In response to increased needs the database system has been redesigned.

The new database environment, as described in this paper, consists out of two new concepts:

- A *Palantir* which is a per machine process that stores the locally declared data and forwards all non local requests for data access to the appropriate machine. It acts as a storage device for data and a looking glass upon the world.
- *Golems*: working units that define the data within the Palantir, and that have knowledge of the hardware they control.

Applications access the data of a Golem by name (which do resemble Unix path names). The palantir that runs on the same machine as the application handles the distribution of access requests.

This paper focuses on the Palantir concept as a distributed data storage and event handling device for process control.

I. INTRODUCTION

The National Institute for Nuclear- and High-Energy Physics (NIKHEF) operates for about ten years a linear medium-energy electron accelerator (800 MeV). Currently the accelerator is extended with a pulse-stretcher ring. At the same time the experimental facilities are renewed and according to the plans the first experiment with this new set up will start in the summer of 1992.

All these facilities are controlled by a number of computers running a home made real-time operating system in a point to point communication network and a number of Unix based machines (ref[1]). To prevent possible conflicts all hardware control is performed under the supervision of a device, which historically is called a database, in which all values are stored before they are sent to the hardware. This database was designed ten years ago, uses one (real-time) machine as central storage device. Although this system is functioning well and has proved its reliability in the past years, it has some disadvantages: it offers two different way to access data (by name, and by number), changing its layout is cumbersome and maintaining it appeared to be quite difficult.

Obviously, we wanted something new. This paper describes the aims, the concept and some details of the implementation of the new system. In the last section the current status of the project and plans for the future are discussed.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S09DPP02

II. AIMS FOR THE NEW DATABASE SYSTEM.

Because we did built the current system ourselves, used it and maintained it for the past ten years, we had enough experience to define the aims for the new system. As the database system is meant for real-time, on-line process control it is clear that it should be fast enough to meet the needs for this kind of applications. Furthermore it must be possible to use the system in a heterogeneous environment, consisting of as well as real-time systems as Unix systems.

For the database itself we agreed that the new database should:

- Not duplicate data, i.e. data is only stored in one place to avoid inconsistencies.
- Be distributed without applications being aware of the distributed nature of the database.
- Be flexible and easily extendable.
- Not be aware of its intended use; i.e. not have any knowledge of the hardware it is intended to control.
- Offer as services at least: read, write, lock and subscribe.

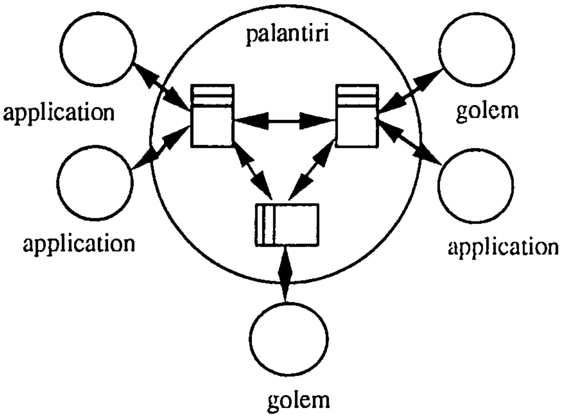


Figure 1. The Palantir database concept.

III. WHAT DOES THE NEW DATABASE LOOK LIKE ? PALANTIRI AND GOLEMS.

Palantiri [Quenya] 'Those that watch from afar', the seven Seeing Stones brought by Elendil and his sons from Númenor; made by Fëanor in Aman.

J.R.R. Tolkien, The Silmarillion, George Allen & Unwin, 1977, P 346.

Golem *n.* clay figure supernaturally brought to life (in Jewish legend); automaton, robot.

The Concise Oxford dictionary, 7th ed. 1982, P 426.

(These new names were invented to avoid confusion with the terms used in the existing system.)

On each machine that requires access to the database runs one specific process; not a server, not a daemon, but a *palantir*. This is indeed what the name suggests (to those familiar with Tolkien): a crystal ball giving access to all other *palantiri* in the system. On top of that, palantiri are able to store data that has been declared by their local agents/clients, which we call *Golems*.

A *Golem* is a process that typically, but not necessarily controls some hardware. The Golem has knowledge of this hardware, and receives the values to steer the hardware from its local Palantir. All relevant data that must be known to the outside world are sent to the Palantir by the Golem.

For the third type of process in our system we have not chosen a new name; processes that communicate with Palantiri, but do not own variables, are called Applications.

A Golem has a name which is made known to the Palantir, and owns a set of variables which are also made known to the Palantir. The Palantir allocates memory for these variables and maintains their values.

Variables have a name, a type and access rights attached to them, all of these are specified by the Golem. The variable name looks like a Unix path name. The first element is the name of the golem, all other elements are given by the golem. But there is a direct relation between the variable name and the structure of the data. The variable called *should* in the following C-structure, that has been declared by a Golem with the name *A13*

```
struct magnet
{         long is;
          long should;
} Q1;
```

may be accessed by the path */A13/Q1/should*. The *magnet* structure can be read at once by accessing */A13/Q1*.

Access rights are different for the golem that owns the variables and for the rest of the world. The access may be any of read, write, lock and subscribe. Now, read and write will be clear; lock will at least give some idea, but what does subscribe mean?

Any application or Golem that has an interest in the value of a variable (either simple, or compound), can subscribe to this variable. This means that whenever the variable is written to a report is sent to each subscriber stating the name of the variable and the new value. This allows our magnet steering Golem to get a report of the new *should* value of its magnet without continuously polling the Palantir. Actually, it is possible to specify that reports are to be sent only when a write action results in the data being changed.

A lock is a flag that can be set on a variable, again simple or compound, reducing write access to that variable to the locking process. It may be used either to keep data unchanged over certain actions, or to keep all other processes from meddling with variables you are assumed to have full control over. A lock exists until it is removed by the locking process.

IV. A BIT MORE DOWN INSIDE A PALANTIR.

A Palantir is a process that forever waits for a message to arrive. Each message is processed in order of reception. depending on the type of request and the name of the variable with which the request is concerned the Palantir decides to handle the request locally, or to forward the request to another Palantir. Requests that can be handled locally are fully serviced before looking for a next request. Forwarded requests are maintained in a number of queues for future reference when the response from the remote Palantir arrives.

All this is quite straightforward, and Palantiri would be very simple indeed but for exception handling. Machines may become unreachable through network failure, or because the machine itself is down. There is no simple distinction between the two. Or a process having an outstanding lock may disappear, thereby creating a possible deadlock situation.

Palantiri have an elaborate 'are you there' mechanism, both to all other Palantiri and to their local Applications and Golems. An application or Golem that does not exist anymore results in all its locks and subscriptions being removed. Note, that a Golem's variables will remain valid, though any action performed on them will result in a warning message 'Golem dead' to the requester. The dead of a Golem is also reported to all processes having subscribed to any of the Golem's variables. When the Golem comes alive again (this is possible) the subscribers will be notified again.

When a Golem, and its data, is no longer needed, the Golem may be removed. All processes having a subscription or lock on the Golem's data are notified of the removal of the data, and the subscription or lock ceases to exist.

When a Palantir becomes unreachable, again all subscribers to any variable on that Palantir are notified. Locks on that Palantir stay in effect, but obviously, any action to that palantir must result in an error message. When the Palantir becomes reachable again, it is possible (well usually) to differentiate between network failure and machine failure. When the unreachability was the result of network failure, it is assumed that locks and subscriptions are still valid. When the remote machine has been down, locks and subscriptions are reestablished as soon as the Golems they are concerned with come to life again.

Other nice problems came into existence by requiring Palantiri to be transparent. In a homogeneous environment this is no problem, but we do have computers of different types, each having its own data representation. Several solutions were considered, but in the end data is stored in the Palantir's local format, and is sent over the network in the requesters local format. This has one large disadvantage: When a new type of machine enters our environment, we will have to recompile all Palantiri after adding the appropriate conversion routines.

V. CURRENT STATUS AND PLANS FOR THE FUTURE.

The implementation of Palantiri on the Unix systems has been completed. They have been tested under various conditions and seem to be running fine. Documents are available with the functional specifications of the Palantir itself and the Palantir access library (ref [2,3]).

In a makeshift setup though, performance tests have been done, and even in a situation with many variables (20000) spread over 200 Golems running on few machines (4) and a high load in subscription reports (>100/second) the system kept running nicely.

As a nice side effect, it seemed possible, even simple, to create a C shell environment in which the entire Palantir database can be accessed very much like the Unix file system tree. Hence we now have commands like *pls*, *pcd*, *pget* and (slightly more difficult) *pput*.

The control system of the new experimental facilities will be based on the Palantir concept.
The control system for the linear accelerator and the stretcher ring is based upon the 'old' database concept. Porting this system with all applications involved to the Palantiri database is a large project. Because of operational and manpower aspects, it is impossible to do the port in a short period of time. To be able to convert and test existing applications, golems have been made that map the existing control database onto the new Palantir domain. Doing this, the two domains are connected and it is possible to convert first all applications and then replace the 'old' database by the new one.

VI. REFERENCES.

[1] W.P.J. Heubers and R.G.K. Hart, A Workstation-based Operator Interface to Control MEA. Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems, Vancouver 1989.

[2] B.J.Tummers, Palantir Functional Specifications, internal NIKHEF document.

[3] B.J.Tummers, Palantir Library Functional Specifications, internal NIKHEF document.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S09DPP03

Intelligent Trigger by Massively Parallel Processors for High Energy Physics Experiments

Francois Rohrbach
CERN, CH-1211 Geneva 23, Switzerland
G. Vesztergombi
Central Research Institute for Physics (KFKI), H-1525 Budapest 114, POB 49, Hungary

Abstract

The CERN-MPPC collaboration concentrates its effort on the development of machines based on massive parallelism with thousands of integrated processing elements, arranged in a string. Seven applications are under detailed studies within the collaboration: three for LHC, one for SSC, two for fixed target high energy physics at CERN and one for HDTV. Preliminary results are presented. They show that the objectives should be reached with the use of the ASP architecture.

I. INTRODUCTION

High luminosity hadronic colliders (LHC & SSC) will require novel detectors, both highly time-sensitive and selective. Potentially, Megabytes data will be produced at rates (66 MHz at LHC) that are beyond performance of today modern transmission and recording technology. From this huge amount of information, however, only a tiny fraction is possessing any real interest. The required high selectivity is assumed to be achieved by a two steps procedure. A first-level decision based on simple "hard-wired" logics can provide significant rate reduction, it leaves, however, for the second-level decision so complex patterns which require a detailed analysis similar to what is done today in off-line programmes, but with an event frequency of typically 100 kHz. Such decisions, based on a huge number (10 to 100 Mbytes) of digitised local or global data coming in a narrow time window, will require the fast execution of precisely tuned algorithms in extremely fast computer-like devices. Industry and computer science make serious efforts in this field. The MPPC (Massively Parallel Processing Collaboration) is concentrated on problems that are likely to benefit from massive parallelism of SIMD type. Such massively parallel machines operate with thousands of processing elements, all highly integrated and controlled under a single controller. Taking advantage of the application needs and of the coincidence between technological opportunities - the development of a new kind of SIMD machine by ASPEX Microsystems (UK): the ASP (Associative String Processor [1]) and the continuous improvement in silicon integration (VLSI/WSI) - a Research and Development programme "The MPPC Project" has been launched [2-5] between ASPEX (UK), CERN (CH), CEA/CEN-Saclay and CNRS/IN2P3-LAL-Orsay (F), as main partners, and EPFL-Lausanne (CH), University of Brunel (UK), University of Geneva (CH), CRIP/KFKI Budapest (H) and Thomson-TMS (F), as associated partners. The applications are dominated by but not exclusively driven by the problem of triggering events in HEP; EPFL, as MPPC partner, is indeed working on a first application in image processing for HDTV. More generally, it can be expected that the same basic processing elements will find their way into quite different application fields. Indeed, the almost infinite scalability of the ASP architecture[1] and its impressive performance targets (in terms of cost, power and achieved density) will attract other suitably parallelized projects (e.g. relational data processing, simulation, computer vision, cellular automata, neural networks) in applications such as, high-definition TV, autonomous guiding vehicles, artificial intelligence, medicine, space science, meteorology, plasma physics, etc. Even one can think about possible application for on-line accelerator control.

II. THE ASP ARCHITECTURE

The choice of the ASP, as a R & D platform for the Collaboration, was based on the exceptional potentialities offered by this new architecture which allows a wide range of applications.

The main hardware task is to build four ASP machines, one for each main partner, with 16384 APE array, referred to as the "MPPC array". It is based on the existing VASP-64 VLSI ASP chip used for the TRAX-1 machine, another ASP project dedicated for off-line image processing [6,7]. The MPPC-array design allows for maximum processor element density and maximum direct parallel interfacing via conventional electronics to the readout of particle detectors. For this task, dense packages of ASP must be constructed. This is based on a modular design, using hybridation on insulators of the VASP-64 chips. These modules are built by PolyCon (USA) ; they contain a string of 1024 APEs (16 chips) with two parallel I/O per module. These modules will be installed on boards to make 8K strings. Two ASP boards and a low level controller (LAC) in extended VME standard (in order to be compatible with existing industrial modules) are under construction for each 16-K MPPC-Array machine.

A. *Associativity and string features*

The ASP consists of associative processing elements (APE) working as SIMD machines. The string can be arranged in a loop: the architecture is reconfigurable by programmation. Each APE is an associative memory cell with processing capability. Synchronous and asynchronous communication between APEs is provided through an inter-APE communications network dynamically reconfigurable. APEs are addressed by content through a common bus, which minimises data movement. Parallel processing is performed on active APEs selected at a given step of the programme. The architecture is scalable up to hundred of thousands APEs, due to high integration (VLSI/WSI) and low power consumption (~ mW/APE). The low target cost (below 5.- SFr per APE) is leading, together with the low power and high integration capability, to the possibility of massive integration. The system has maximum application flexibility and computational efficiency. It is fault tolerant: blocks of faulty APEs may be disactivated without breaking the string . The MPPC-Array machines will have Parallel I/O capabilities (10 Gbit/s). ASP application programmes can be written in any block-structured language (Modula 2 is the commonly used language). An introductory course to ASP, provided by ASPEX, is useful to reach a good level on the learning curve in parallel algorithms.

B. *ASP chip and module*

The basic chip for the construction of the compact hybrid MPPC modules (HASP) consists of a programmable VLSI SIMD parallel processing device, incorporating 64 associative processing elements (APE, see fig.1 and 2): the Aspex Microsystems Limited VASP VLSI ASP chip presently manufactured with $2\mu m$ technology at ES2 (F).

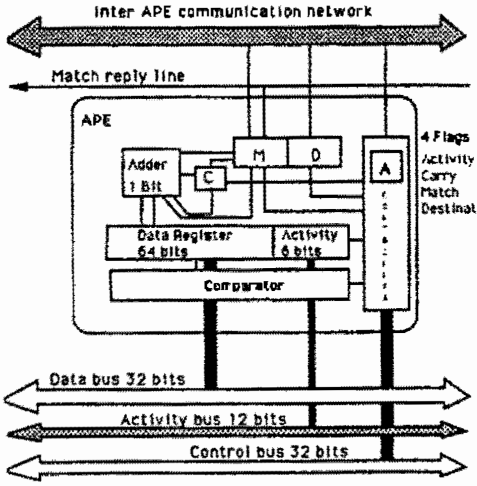


Fig. 1 - Schematic of the associative processing element

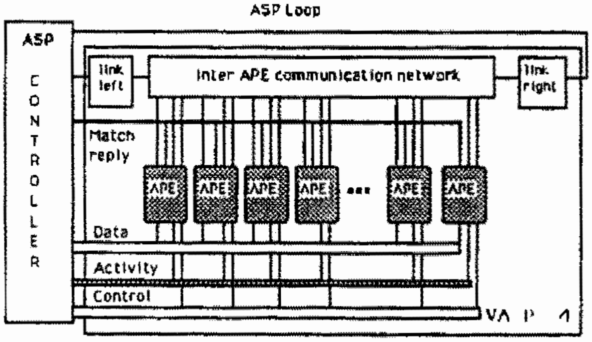


Fig. 2 - The basic Associative String Processor

This chip, although slower than expected, is suitable for making the first prototype hybrid HASP module . This module is under development at ASPEX for the MPPC-Arrays and is to be manufactured by the sub-contractor Polycon Inc. (USA). For second-level triggering experiments a faster device will be required. It will use a $1.2\mu m$ SOS (silicon-on-sapphire) device in order to achieve a fully working 25ns VASP chip in the summer of 1991 (a high performance ASP chip presently developed by ASPEX and Hughes Company). It will be the basic stone for the final hybrid ASP module for MPPC-arrays: the HASP/P1. This 1K APE module, using 16 dice, has a bypass of 64 and 256 APE blocks and 2 I/O ports which can be configured as 2 x 512 APE substrings or a 1K substring with a LAC interface and an ADB (ASP data buffer) interface. The design is targeted to a standard 184 pin package (3" x 3" with leads).

The operating system and programming tools are ready for a test using the controller (LAC prototype) under construction at Saclay.

C. *Machine architecture*

Each MPPC-Array will be composed by one LAC and two ASP boards giving a 16 K machine. In concentrating all ASP boards in the same machine we will have the 12-bit addition). The ASP board will use the hybrid modules. Each ASP board will contain 8 modules giving a number of 8192 APE per board. Each module will have its own data exchange and ADB double port memory to allow a faster feed for data

III. APPLICATIONS: STATUS AND FIRST RESULTS

As previously stated, seven applications are studied: three LHC oriented, one SSC oriented, two for fixed target physics at CERN and one for HDTV.

A. *Muon selection at LHC (CERN)*

At LHC single and multi-muon triggers will play a crucial role in particular for Higgs search through its four leptons decay. An analog first-level muon trigger is expected to cut down the single particle rate below 10^5 Hz. In order to identify the muons, a more sophisticated second-level trigger is further required. This includes momentum determination, charge assignment and counting all the tracks which are above a given cutoff momentum. This task must be achieved in less than $\sim 20\mu s$ in order to cope with the rate of the first-level trigger. The solution which is under study proposes to divide the triggering procedure into three phases: the loading of the hit information coming from the muon detector (detector mapping into the ASP), the preprocessing of the data (determination of the best hit positions: the "Master point determination") and the tracking (track finding, charge signature and p_T determination).

Taking advantage of the rotational symmetry of the CMS detector model [8], a (r, ϕ) mapping of the fine-grain muon detector can efficiently be done in the ASP [9]. A track is defined in the central plane of projection (plane of deflection perpendicular to the beam axis) by the vertex of the interaction, positioned with high precision, and by the muon trajectory detected at five radii with multilayer detectors . The mapping is done in a way which associates one APE to each $\Delta\phi$: this allow to identify in a single search all the high p_T tracks. The granularity $\Delta\phi$ (typically in the mrad range) is a parameter dictated by the size of the Coulomb-scattering. As long as $\Delta\phi$ is above its minimum physical detector pitch value, the mapping is reconfigurable according to the chosen triggering p_T threshold.

Only the active cells are loaded into the ASP, preserving the topology of the detector which provides the so-called "iconic mapping". Then, a "master-point" is calculated from the multi-layer detector hits, taking into account detector inefficiency and possible multiple hits in one or more layers. This preprocessing is done using a fast "iconic average" using data shifting and bit-logic operations across the APEs. For the third step of the triggering scheme, the tracking, iconic algorithms use bit representation of the image of tracks and the image processing can be done bit parallel in the ASP machine. The five consecutive values of the azimuthal angle ϕ relative to the innermost ϕ_0 value, expressed in unit of $\Delta\phi$ is used for calculating a "track-code ". The track-code is a unique representation of a given charged particle trajectory through the muon detector. A preliminary study was to construct track codes from the Geant Monte Carlo data and to explore the feasibility and efficiency of triggering by the proposed iconic algorithm. It consists of looking for all hits at the same time for a fit between master points configuration and all possibilities of track-codes. The number of track-codes depends on the ϕ binning. This number determines the muon search execution time. On the contrary, the triggering time will not depend upon the muon multiplicity because the search is done in parallel over all ϕ_0 and ϕ values. For triggering purpose ($p_T \geq p_T$ threshold) a momentum is determined for each track, by assigning to each track-code a maximum p_T value using a simple look-up table. The preliminary results of the simulation show that the trigger can be worked out within about 20 μs: 5 μs for loading, 5 μs for preprocessing and 10 μs for tracking. This timing fits the requirements imposed by the expected first level trigger rate at LHC.

B. *TRD electron selection at LHC (CERN)*

An integrated transition radiation detector (TRD) and charged particle tracker has been proposed for a LHC detector in order to improve the identification of electrons beyond the level of electromagnetic calorimetry[10]. The TRD tracker will have about 500 000 channels (straws) put inside a cylinder installed around the beam axis (the Halo model). Electron candidates will be tagged by a surrounding calorimeter and the information will be used to define planes cutting the tracker detector and defining candidate roads inside the TRD. One assumes limits on both, the number of candidates (no more than four per event) and the rate of candidate occurrence (not more frequently than every 10 μs average). The basis of discriminating electrons from hadrons in the TRD tracker lies in the statistical analysis of pulse heights of all digitising belonging to a track candidate. This allows to measure the probability of TRD X-ray emission (and detection), which is strongly enhanced for electrons due to their very high γ values. As for the muons, the trigger procedure can also be divided into three phases, but each of them is being implemented on a different type of hardware for optimising the running of the dedicated algorithms.

C. *LHC calorimetry, jets and shower detection (Orsay)*

The use of ASP at the second-level trigger of a barrel calorimeter model for LHC is under detailed study using the ASP simulator. It is assumed that the event buffering at that level should not exceed 100 μs on average. Special care is put on the study of the mapping of the calorimeter cells into the ASP (patching optimisation). The basic procedure is to associate one APE to each cell. The loading time will be of the order of 10 to 15 μs. If the Vector Data Buffer feature (VDB: a word parallel, bit serial ASP loading under development at ASPEX) becomes available on the chip, most of this time will be overlapped with the processing of the previous event, and the real cost of loading will only be 1 μs. A fast rejection of each event detected inside the calorimeter should be done in an average time of 50 μs. Jet energy, isolated electrons, missing energy, shower shape and position are the essential event feature extracted from the calorimetry. Various selection algorithms are under detailed study. They are based on the analysis of the energy deposited in neighbouring cells, making correlations between the information coming from the electromagnetic and hadronic parts of the calorimeter. The possible use of preshower detector is also considered. As a result, 20 to 30 μs processing time is obtained.

D. Possible use of ASP for SSC/SDC detector (Saclay)

At SSC, the measurement of jet energies is essential for the detection of neutrinos and others unseen particles. The SDC (Solenoidal Detector Collaboration) detector [11] is based on a (ϕ, η) tower segmentation where ϕ is the azimuth angle and η the pseudo-rapidity ($\eta = -\ln\tan\Theta/2$ where Θ is the polar angle). Each tower is logically divided into five layers providing the symbolic information which will give the necessary event topology and $e/h/\mu$ particle identification used for triggering. Starting from the vertex, the five layers of detectors give the following information: tracking (hits/track), answer for isolated electron, number of electron clusters, number of hadron jets and muon hits. In the calorimeter, an isolated electron pixel is characterised by an electromagnetic energy value greater than an e.m.-threshold, a hadronic energy value lower than a hadronic-threshold, and no direct neighbouring cell with e.m.-energy greater than an other e.m.-threshold. The basic principle of using ASP in a second-level trigger is to associate one APE for each calorimeter cell and to load into this APE all the information about the five layers contained within a corresponding (ϕ, η) tower. Inside each APE, the 64 bit data register is enough for storing all this information. For the detection of missing energy, the calculation of the transverse energy E_t is done by summing $E_{\sin\Theta}$, calculated simultaneously in the APEs for each cell. Preliminary results of algorithms simulation using the VASP-Simulator give 7 μs for the detection of isolated electrons and $\sim$ 20 μs average time for missing energy (dependent on clusters number and geometry).

E. A K_0 trigger for NA48, a fixed target physics experiment at CERN (Saclay)

The NA48 experiment is an experiment aiming to perform, in 1994-96, a high precision measurement of the ϵ'/ϵ parameter in order to have a better understanding of CP violation [12]. This parameter is determined from the measurement of charged $(\pi^+\pi^-)$ and neutral $(2\pi^0)$ decays of K_0^S or K_0^L, concurrently. The target is to obtain on-line a very good signature of $2\pi^0$ candidates in less than 10 μs, taking as inputs the energy deposits in the 12000 cells of an electromagnetic calorimeter array. Candidates should give exactly four photon shower clusters in the detector. Physics constraints from the K_0 decay are used for validating good triggers: the transverse momentum conservation implies zero value for the first moment of the energy distribution (relative to the centre of the calorimeter) and, from the K_0 mass constraint, the vertex position can be calculated by using second moment of the energy distribution and total energy of the clusters. These calculations should be invalidated if an accidental hit occurred in the calorimeter in the sensitive time window. Processing time was evaluated for the ASP and for digital signal processors (DSPs). ASP is better suited for the topological processing tasks (find clusters and count them, find accidentals and locate them relative to the normal signal timing), while DSPs are better on the fast, high accuracy, stream arithmetics required in the energy balance and vertex calculations. This application is an example of the use of ASPs in prompt trigger systems, where real time response performance, and fast parallel loading capability is of prime importance. The result of this study shows that combining ASP and DSP processors in the fast neutral NA48 trigger system is currently sufficient to fulfil the NA48 requirements: an efficient $2\pi^0$ trigger can be performed in less than 10 μs.

F. ASP tracking with CCD on-line camera in WA93, a fixed target heavy ios physics experiment at CERN (U. of GENEVA)

A heavy ion experiment (WA93) is scheduled at CERN with the aim to study Bose-Einstein correlations among charged pions. The reconstruction of the pion momenta with a large acceptance will be done with a tracking system consisting of a spectrometer magnet and new type of light emitting multistep avalanche chambers[13]. The passage of a charged particle through the chamber is detected as a cluster of light registered by Megapixel CCD cameras. The analysis implies various stages: image preprocessing (back ground reduction and subtraction, optical distortions) then actual image processing (cluster analysis) and finally tracking and momenta correlation . The use of ASP for these tasks looks very promising as they are well adapted for massively parallel image processing. Bose-Einstein correlation requires computation of four-momentum difference for all pairs of tracks which means that many TFLOPS of computing power will be required for only a few days of running ($\sim 10^6$ events).

G. Image sequence coding, data compaction for HDTV (EPFL)

At the signal processing laboratory (LTS) in EPFL, the image sequence coding group has developed number of techniques for image sequence compression reaching very high compression ratios. For real time processing at video rate of large images, a parallel computation approach is necessary. This is why the use of ASPs is studied using two different methods: the parallel implementation of the

Gabor compression algorithm [14] and the parallel implementation of artificial neural networks on the ASP [15]. In the Gabor compression algorithm, the image is decomposed in elementary Gabor functions basis. The use of such a decomposition is motivated by the fact that Gabor functions have optimal localisation in both spatial and frequency domains. The realisation of this algorithm requires the study of efficient parallel algorithms for matrix computation, particularly large matrix multiplication (typically 256x256). Several algorithms for matrix multiplication on ASP are under study. At this time, two programmes for integer computation have been developed, one using 8 bits and one 16 bits. The best results have been obtained with an algorithm called outer product [14]. The initial results obtained with the ASP simulator in function of the matrix dimension for 8 and 16 bits integers are very encouraging in view of video rate compression up to 200:1 ratio.

IV. CONCLUSION

From the applications under detailed studies for high energy physics at the future hadron colliders LHC, SSC and for SPS fixed target experiments at CERN, preliminary results obtained from simulations, based on the use of ASP machines developed in the MPPC project, can be summarised as follows:

- a second level muon trigger at LHC is feasible and could be done within about 20 μs.
- a second level trigger for calorimetry at SSC and at LHC would require something like 50 μs,
- for the NA48 fixed target experiment, a $K^0 \rightarrow 2\pi^0$ trigger could be achieved in less than 10 μs.

All these results are encouraging and could fulfil the ambitious objectives of these applications. In another domain, for HDTV and videophone applications, preliminary results for compression and restoration of images show that using ASPs could open the possibility of working algorithms at video rates. Real time tests for the seven applications studied at present should be possible in about a year, when the four MPPC-array machines become available.

REFERENCES

1. ASP: A cost effective Parallel Microcomputer, R.M. Lea, IEEE Micro, October 1988.
2. The Massively Parallel Processing Collaboration, The MPPC project MPPC/89-1, CERN, 30 June 1989, Proc. of a Meeting held at EPFL Lausanne 9 June 1989.
3. Proposal- The MPPC Project, MPPC / 89 - 2, CERN, 28 July 1989.
4. Memorandum of Understanding for the Execution of the MPPC Project, CERN/EF/FR/fr, 16 March 1990.
5. MPPC -1990 Status Report, CERN/DRDC/90-76, DRDC/M4-Rev., 18 January 1991.
6. TRAX-1 proposal, Proposal to develop a physics image processing workstation for Megabytes per event data based on fine and medium grained parallel processors, A. Sandoval et al., CERN, 27 April 1988.
7. CERN-LAA/89-1, 15 September 1989, section II.6, p.267.
8. Study of muon triggers and momentum reconstruction in a strong magnetic field for a muon detector at LHC, CERN/DRDC/90-36, DRDC/P7, 30 August, 1990.
9. Second-Level Muon trigger for the Large Hadron Collider, G. Odor, F. Rohrbach, G. Vesztergombi, CERN/ECP 90-20, 21 December 1990. Aachen LHC Workshop Proceedings CERN 90-10, vol.III. p.136.
10. Integrated High-Rate Radiation Detector and Tracking Chamber for the LHC, Proposal, B.Dolgoshein, spokesman, CERN - DRDC/90-38.
11. SDC, Letter of intent SSC/LOI-0001, 30 November 1990
12. Proposal for a precision measurement of ϵ'/ϵ in CP violation $K^0 \rightarrow 2\pi^0$ decays CERN/SPSC/90-22 SPSC/P253, 20 July 1990.
13. Proposal for a Light Universal Detector for the Study of Correlation between Photons and Charged Particles CERN/SPSC 90-14 May, 1990 and CERN/SPSC 90-32 October, 1990.
14. T. Ebrahimi, T.R. Reed, M. Kunt, Video coding using pyramidal Gabor expansion, SPIE, Vol 1360, Visual Communications and Image Processing 90, p. 489-501.
15. G.W. Cottrel, P. Munro, D. Zipser, Image compression by back propagation: an example of extensional programming, ICS report 8702, February 1987, UCSD.

For more details on ASP we are available by e-mail: ROHRBACH @ CERNVM.CERN.CH and VESZTER @ CERNVM.CERN.CH.

Realtime Aspects of Pulse-to-Pulse Modulation

Rudolf Steiner, Claus Riedel, Werner Rösch
GSI, Postfach 110552, D-6100 Darmstadt, Germany

Abstract

The pulse-to-pulse modulation of the SIS-ESR control system is described. Fast response to operator interaction and to changes in process conditions is emphasized as well as the essential part played by the timing system in pulse-to-pulse modulation.

I INTRODUCTION

The benefits of pulse-to-pulse modulation in acceleration operating have been described as early as '77 [1]. It is an effective way to increase the overall output of valuable beamtime of one or more accelerators. With beamsharing, rarely all users of the beam will be unable to accept the beam at the same time. If the PPM-handling quickly responds to changing conditions, there will be virtually no dead-time in the machine operating due to inevitable dead-times of experiments, e.g. during new experimental setups.

In a multi-accelerator facility, PPM is almost imperative. Asynchronously running machines, every one of them operating as an injector for the next one, normally have time left between subsequent injections that can be used for experiments.

II CONTROL AND TIMING SYSTEMS

Much has been said and will be said at this conference about the major trends in control systems in the last decade. Most systems recently designed or upgraded are looking more and more similar: Graphic workstations are the operators' I/O tools and software development platforms. The workstations are linked to a communication backbone. Ethernet, Token Rings, or fiber optic links are candidates. The choice rather depends on the distances to be mastered than on technical advantages or disadvantages.

To the same backbone, eventually as subnetworks with bridges in big systems, the equipment control computers are connected, which are mainly VME-, Multibus-, or CAMAC-based.

Extensive use of graphics and CASE tools has made an essential improvement in the operator's and software designers' access to control systems.

The overall trend is from very special systems tailored to the very special task of accelerator control towards more uniform, general purpose systems and the use of standards of the marketplace.

On the process level, however, the special needs of accelerator control, mainly realtime and synchronisation, do still exist or are becoming even more complex. Therefore the functionality of a control system must be biased by a timing system. The diversity of control systems of old has it's evolutionary relic in the diversity of timing systems, which will resist standardisation trends for another while. The more the higher levels in a control system become general purpose (and less realtime), the more process-specific problems must be solved on the lower levels. This is the domain of the timing system, the equipment controllers, and, of course, the equipment hardware. In the trend to general purpose systems the design of the timing system determines the overall performance significantly. The functionality of the timing system may range from simply providing clock signals for subsystem synchronisation to sophisticated timing control of the equipment control computers. However: A control system is only complete with a timing system (Fig.1).

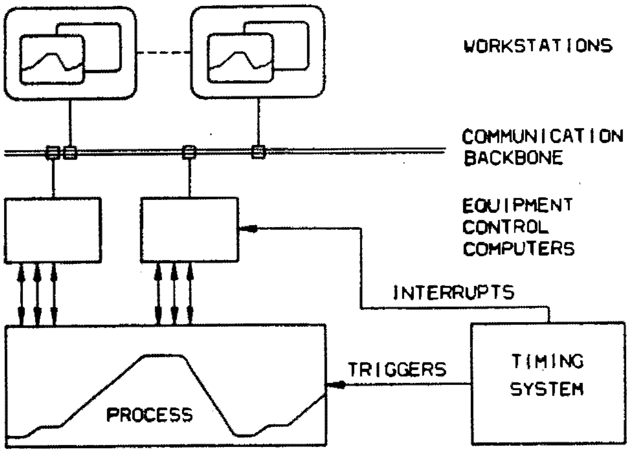


Figure 1: A complete control system

III PPM-MANAGEMENT IN REALTIME

III.1 Realtime Demands of PPM

The control system of Fig.1 already contains all realtime and synchronisation mechanisms to manage the process, e.g. to execute a synchrotron cycle. A system that can execute one cycle can as well execute another one. Nothing principially new needs to be introduced to perform PPM.

What then could be the realtime aspect of PPM? Let me call it the 'online supercycle management'.

There are three levels of access to supercycle management, with increasing realtime demands:

III.1.1 Offline Management

The supercycle is built up offline to fit the experimental program. The internal timing of the individual cycles is programmed at the same time. There is no realtime demand.

III.1.2 Online Management

The operator may have a need to change either the supercycle or the timing of an individual cycle. The presently running cycle continues unchanged, but the next one will have the modifications.

III.1.3 Process Driven Management

Process conditions may change and need fast response (suspend immediately one type of cycle of the supersycle) or even very fast response (emergency, dump the presently running cycle). Evidently this is beyond the operators' abilities and must be handled on the process level.

For fast response, a request mechanism is very useful: Cycles are only executed upon request. This is the appropriate level to feed in additional conditions, as shown in Fig.2. Effective exploitation of beamtime is not the only aspect of PPM. Security, radiation protection etc. are other ones: A high energy beam must not be generated if it cannot reach it's destination point properly.

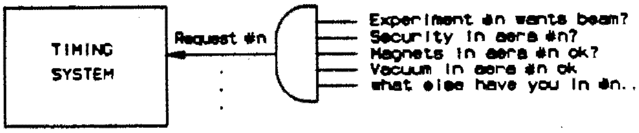


Figure 2: Request Level

Emergency handling is too special in every accelerator environment to be discussed here.

III.2 An Example: PPM Management at SIS-ESR

Fig.3 shows the control and timing system of SIS-ESR. The timing system drives a timing network in parallel to the communication network. Timing interfaces receive the serial timing information and pass it as a 16-bit parallel code, the event-code, to the equipment control computers.

A selectable set of events is transformed to hardware triggers for equipment by the timing interfaces. The time jitter for triggers is smaller than ±.5 microseconds in the system. Nanosecond timing has been left to special solutions to keep the timing system hardware simple.

In contrast to Fig.1 the timing system in Fig.3 is not a special part in the control system. To the operator it looks as, and in fact is, just another equipment control computer.

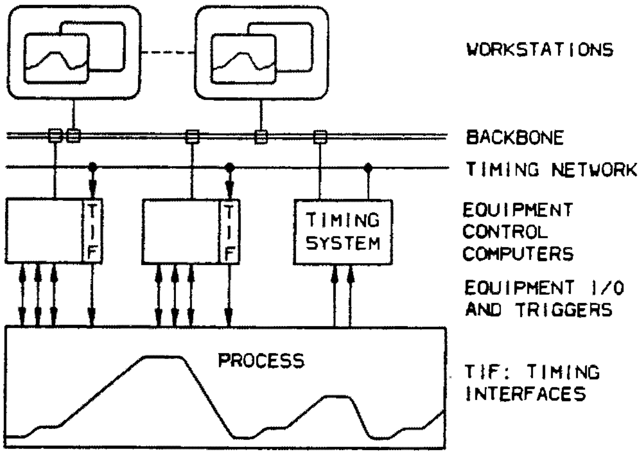


Figure 3: The SIS-ESR Control Architecture

The SIS-ESR timing system has direct timing control over all equipment control computers [2]. Each equipment control computer has available complete settings to execute sixteen different cycles, the 'virtual machines'.

The timing system activates machines by providing the event-code (Fig.4). The equipment controllers do nothing unless they receive events, command events, if nothing else, when no cycle is active and the timing system is idle. In fact, the timing system of the SIS-ESR is a hardware dispatcher to compensate for the loss of realtime abilities on higher control levels: It is the timing system that makes a 'real machine' out of a 'virtual machine'. This given, it is the natural candidate to be the 'supercycle manager'.

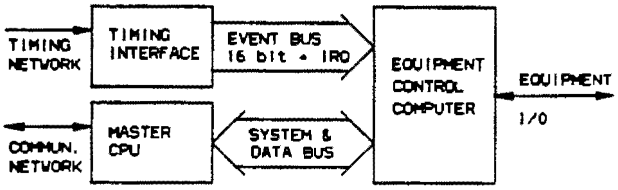


Figure 4: Event-Driven Equipment Control Computer

In comparison to other PPM-solutions, e.g. [3], no special PPM-management components like PPM message decoders are needed at the process conrol level.

Fig.5 gives the operator's view of a simple supercycle.

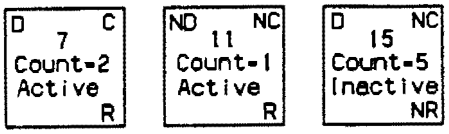


Figure 5: A SIS Supercycle

The properties assigned to each cycle are
active/inactive
execution count
decement/nondecrement
contiguous/noncontiguous

all of which can be changed online by the operator. Active/inactive provides a simple means to suspend a cycle. An inactive cycle does not leave a gap in the supercycle. It continues with the next active cycle.

The execution count works together with (non)decrement and (non)contiguous.

In non-decrement condition, (non)contiguous is meaningless. The subsequent supercycles are identical containing as many individual cycles in sequence as set by the count number.

With decrement and non-contiguous, the cycle will show up once in as many supercycles as set by the count, and suspended in the following supercycles.

With decrement and contiguous, the cycle is executed as many times as set by the count in only one supercycle.

One type of cycle may show up at different places in the supercycle, giving a high flexibility for supercycle programming.

Fig.6 gives an overview of the supercycles resulting from different settings provided the status from the request level is 'true'. If 'false', the cycle in question either will not be executed as in 'non-active' state, or an appropriate gap is inserted into the supercycle, if it is essential to maintain the overall timing structure. The choice is made by a software switch in the timing system.

R/NR is no input channel for the operator. It indicates the request status.

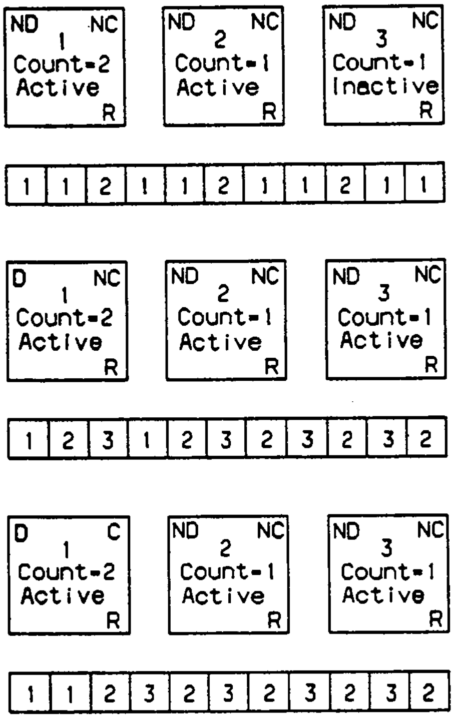


Figure 6: Supercycle settings

The properties given to the cycles, together with the request mechanism, easily allow a preprogrammed dynamic behavior of the supercycle. This is extremely useful when conditions change inpredictibly, e.g. when the storage ring ESR requests a bunch of cycles every few hours.

IV SUMMARY

The benefits of online PPM-management have been described. As an example the PPM-control of the SIS-ESR has been shown. the PPM-management in the SIS-ESR control is very simple and straightforward due to two facts: The central role that has been given to the timing system within the controls, and the proper design of this system. With direct control of the equipment controllers' timing it is the beating heart in SIS-ESR controls, rather than anything else.

After four years of operation, we found no demand for timing that could not easily be granted.

V DEFINITION

A 'virtual machine' in this context means a complete and consistent set of data stored in the Equipment Control Computers to perform an accelerator cycle if activated by the timing system.

VI ACKNOWLEDGEMENTS

[1] The PS Staff, reported by J.P.Potier, Pulse-to-pulse modulation of the CERN PS complex, Particle Accelerator Conference, Chicago, 1977

[2] W.Rösch, R.Steiner, SIS-Timing, GSI Scientific Report 88

[3] Bolliot J. et al., IEEE Trans.Nucl.Sci. NS-28 (1981) 2195

Injection Timing System for PLS

S.S. Chang, M.S. Kim, S.J. Choi, S.C. Won, W. Namkung, and S.Y. Park

Pohang Accelerator Laboratory,
POSTECH, P.O.Box 125,
Pohang 790-600, Korea

Abstract

The ultimate goal of the PLS timing system is to successfully inject a electron bunch to a predesigned bucket in the Storage Ring.

In the Linac, a pretrigger of 102.8 microseconds prior to the Gun trigger may be required to charge the pulsed divces properly and it should be precisely delayed to synchronize with beam pass at each accelerating column.

To inject the electron bunch, which fully acceleated in the Linac, into a target bucket of SR, the injection kicker magents must be energized to provide the appropriate magnetic field. For the sequential filling of the SR buckets, the appropriate timing delays throughout the entire timing system are programmably controlled by operator.

Introduction

The Pohang Light Source (PLS) consists of two individual particle accelerators all working in concert to ultimately produce the high brilliance X-ray to the experimenters: the Linac and the Storage Ring (SR). Both accelerators are composed of various transient devices like an electron gun, the booster and klystron modulators, injection magnets, and etc. In the Linac, an electron bunch is accelerated up to 2 GeV for full energy injection, while the SR accelerate the particle only to replenish any energy loss by synchrotron radiation.

The Linac rf system is composed of a booster station, 11 high-power klystron modulator and 10 pulse compressors. It's operating frequency is 2,856 MHz. In the booster station, the driving system provides the optimum condition of the drive signal for high-power klystron and the phasing system make the phase synchronization between the electron bunch and the accelerating wave. The function of these system is to get the maximum beam energy and minimum energy spread on the basis of a stable drive sytem and correctly phased high-power wave guide network. The typical output pulse length from the booster modulator is 4 usec. The high-power modulator supplies pulse voltage to the high-power Klystron. It can operate in two mode : an acceleration mode and a standby mode. In the acceleration mode, the output power is delivered to the accelerating column at the time synchronized to beam pass and then the beam is accelerated. In the standby mode, the output power is delayed with respect to the beam passage and has no effet on the beam. Typical specifications of the modulator are 60 pps pulse repetition rate, and 4.4 us flat top pulse width. Meanwhile, the klystron that utilized the pulse compressor require a phase reversal gate. So the PSK on the booster station is used to reverse the phase of 180°. All of these systems are operated very closely through the timing system.

Injection into the SR accomplishes via a Lamberson septum and the kicker magnets. Prior to injection, the storage ring closed orbit is bumped close to the end of the injection septum by four bump magnets. A pulse of electrons is then transported through the injection septum into the SR. After injection, the bump magnets are turned off in a time corresponding to about two orbits of the ring to prevent the stored beam from disturbunce of the kicker field. The newly injected beam then undergoes coherent betatron oscillation about the closed orbit-motion that is rapidly damped by means of synchrotron radiation damping. This process is repeated at the cycle rate of kicker magnet until the desired beam current is reached.

Timing Sequence

The timing sequence is shown in Fig.1,2 to illustrate that many events are initiated by a pretrigger. In this sequence all the transmission delay are neglected for simplicity. It is supposed also to be no delay for the beam from the gun to injection point, where injection point means the bucket into which the electron bunch is transferred. The time between consecutive buckets passsing a fixed point on the SR is 1.9996 nsec because the SR has 468 stable rf buckets into which a bunch of electrons can be injected. One periode of the rf is named a "tic" so a tic

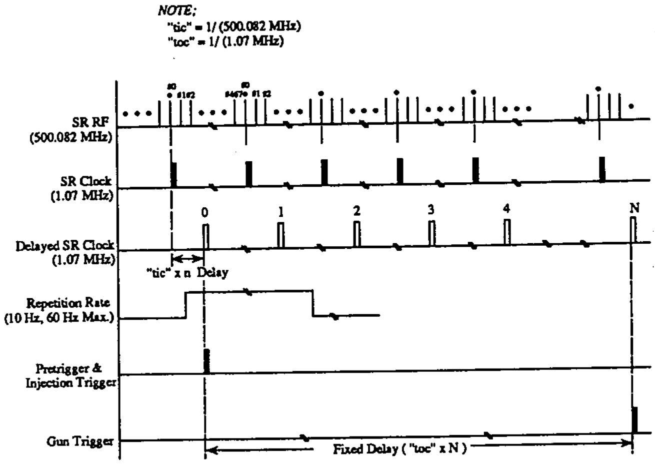


Fig.1 Timing Sequence (I)

is equal to the time seperation between two adjacent rf buckets. When a arbitrary standard revolution clock is defined, each bucket can be uniquely identified by the time displacement of integer multiple of a tic from the standard revolution clock and can be named as bucket #0,#1,...#467. Actually the standard revolution clock is made dividing the rf frequency by the SR harmonics and then the physical bucket #0 is identified as the bucket which synchronized with standard revolution clock in the injection point. The physical bucket #0 is also defined as the bucket filling firstly for the ring operating cycle. To fill the next bucket, for example, as bucket #n the phase of standard revolution clock is shifted a degree of n times of a tic and then the gun trigger is fired at time when synchronizes with this delayed standard revolution clock. On the other hand, A pretrigger is provides 102.8 usec prior to gun grid trigger to properly charge the modulators. Basically, a trigger signal generated on the first edge of revolution frequency falling with a gate drived from the 120 Hz zero-crossing in the repetition rate generator produces a fiducial which is superimposed on the 1.07 MHz singal in the synchronizer is then transmitted on the timing local station and SR kicker station.

In the Linac, a pretrigger is delayed according to the timing requirement of each pulsed devices. Their events are as follow :

1. Gun Trigger ; This signal is used as the reference for all of the other events. The occurence of this signal is determined by the entire timing system and is synchronized to the SR rf.
2. Early Modulator Tirgger ; The high-power modulators must begin charging before any other events.
3. Thyratron Trigger ; This event enable the klystrons. After the modulator rise time, rf begin filling the accelerating waveguides and the pulse compressor cavities.
4. Booster Modulator Trigger ; This event enable the booster modulation amplifier.
5. PSK trigger ; The klystrons used the pulsed compressor need a phase reversal gate. This starts at 0.86 usec prior to the gun trigger signal.

Timing System Description

The PLS timing system consists of a master oscillator, a bucket selector, two rf drive stations, a kicker drive station and other miscellaneous modules. A block diagram of the system description of PLS timing is shown in Fig. 3. The 500.082 MHz rf reference signal is provided to each component via a temperature stabilized line from the main oscillator located in the rf room.

In the bucket selector, the synchronizer generates the pretrigger and a gun trigger, which are synchronized with the SR rf. A gun trigger singal is transmitted to the gun grid pulser without extra delay, in order to reduce the jitter of beam with respect to the ring rf. The pretrigger is split and delayed individually by three kinds of delay circuits to adjust the timing of each modulator.

The repetition rate generator decides an operating cycle to the pulsed devices. Although the pulse repetition rate is limited by the hardware constraint of the power supply of kicker magnet, the repetition rate is variable up to 120 Hz.

Bucket Selector

The function of bucket selector is to make the Linac triggers synchronize with ring rf and to facilitate the selection of an arbitrary bucket. The bucket selector is composed of a ring revolution frequency generator, a synchronizer and a fine delay module.

Fig.4 shows the schematic layout of the bucket selector. Firstly, the standard revolution clock is obtained as divding the ring rf frequency by the ring harmonics. The next, the standard revolution clock is delayed and then the fine delay module is used as a revolution frequency phase shifter. Threrfore we can select an arbitrary bucket by the setting of a preset value corresponding to time displacment of a target bucket. The synchronizer generates two kinds of synchronized triggers : the injection trigger

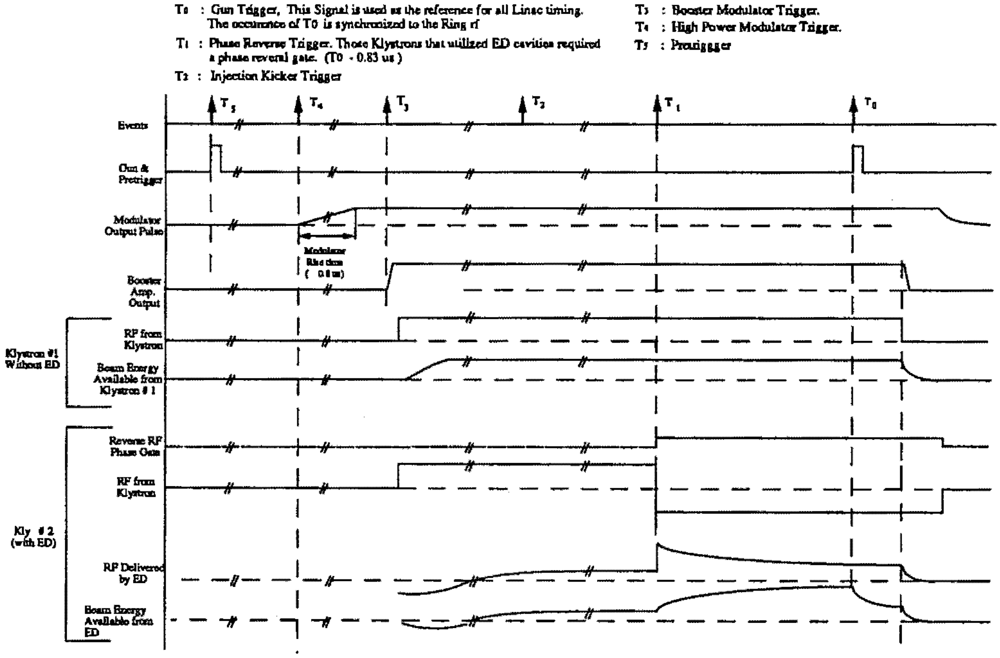


Fig.2 Timing Sequence (II)

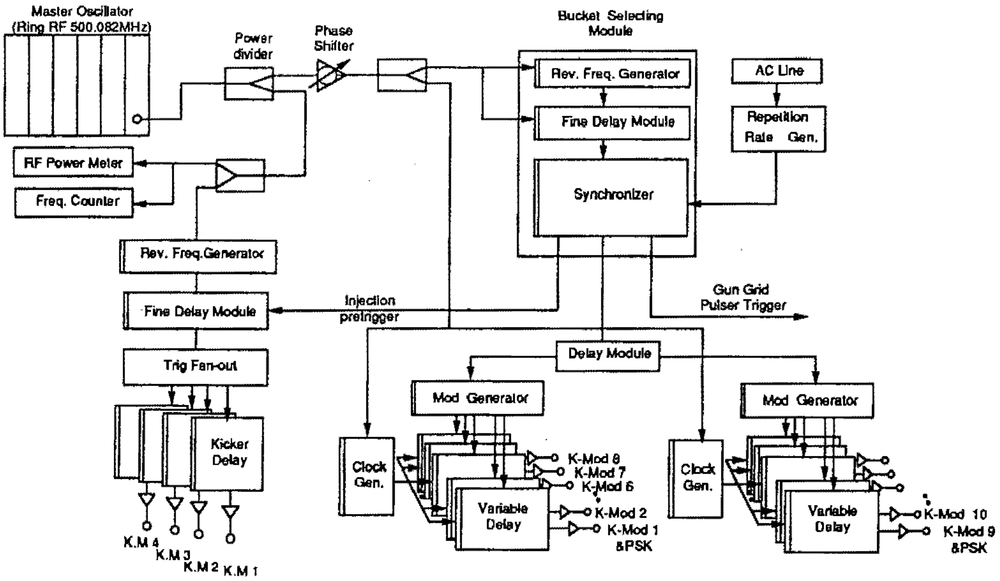


Fig. 3 Schematic Diagram of the PLS Timing System

and gun grid trigger. The jitter of the gun grid trigger output of the synchronizer is less than 100 psec with respect to the SR rf.

RF Drive Station

The rf drive station is designed to adjust individually the trigger timing of each Klystron modulator and a booster modulaor, and to select the trigger mode of each Klystron modulator. At each station it provides 6 mode-selected Klystron modulator trigger signals delayed individually. Each unit of the local station consists of a trigger mode selector, six 10 us/step variable delay modules and a clock generator. The input trigger generates two type of signal : an acceleration mode trigger and a standby mode trigger. Both are fed to the variable delay module for the one of them selected by the mode selector is fed to each Klystron variable delay module.

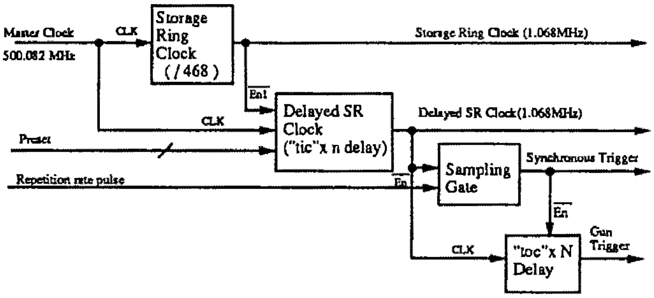


Fig. 4 Schematic Diagram of the Bucket Selector

Kicker Drive Station

To inject the bunch into the SR, the injection kicker magnet must be energized to provide the appropriate magnetic field. The power supply of PLS injection kicker magnet is specified to provide a half sine wave with a base width of 4 usec. Since magnet field must be stable when the bunch arrives, the magnet should nominally be triggered at 2 usec prior to the arrival of bunch at injection point. The kicker drive station is composed of a fixed and some variable delay modules and supplies the pulsed magnets with the delayed trigger pulses according to delay time of each pulsed manget. The lengths and heights of a delayed pulse should be adjusted to drive the pulse magnet.

Control of Timing System

For the high flexibility and reliability of the timing control, VMEbus based computer are introduced and all the timing modules are installed in VME crates. Overall control sytem for the PLS timing is composed of a master and 3 distributed slave system. The master VME system consist of a motorola 32-bit microprocessor, local memories, a Ethernet controller, some I/O controller. The real-time opeating system such as OS-9 is installed and all of the timing data is set and recorded.

References

1. T. Urano, K. Nakahara, Nucl. Instr. and Meth. A247(1986)226.
2. G.R.Aiello, A. Carniel, Sincrotron Trieste ST/M-89/11

Automated Control System Structure of the USSR Academy of Sciences Kaon Facility

Yu. S. Ivanov, V. A. Konovalov, B. P. Murin, L. L. Filipchicov
Moscow Radiotechnical Institute
of the USSR Academy of Sciences
S. K. Esin, Yu. V. Senichev
Institute for Nuclear Research
of Academy of Sciences of USSR

1. Introduction

Up to date at Nuclear Research Institute of the USSR AS (Moscow-Troitsk) it is finished building of Moscow Meson Facility high intensity current proton Linear Accelerator (LA) (beam parameters: energy - 600 MeV, average current - 0.5 mA, pulse current - 50 mA). The LA proposed to serve as Kaon facility (KF) injector which is under working out [1].

Kaon complex, in addition to LA, includes: buster proton synchrotron (BR) with output energy 7.5 GeV, main synchrotron (SR) with proton energy up to 45 GeV and storer-stretcher (SS). The KF is proposed to work at 3 regimes.

At first regime SS follows SR and is used as beam stretcher. KF time work diagram is cleaned by Fig.1a. A half of beam pulses from LA and BR is used at ones for physical experiments. At second regime SS is inserted between BR and SR and works as collector (Fig.1b.). At third regime it is supposed to store in SS 4-6 beam pulses with next fast exit to experiment (Fig.1c). The such kind using allows to receive terra watt power level pulses ($8\cdot10^{14}$ particles with 45 GeV energy) with frequency of 1 Hz. There are presented below brief description of KF systems, which are concerned of radiotechnical systems (RTS) control (ACS) and adjusting (AAS).

2. RTS Interaction

KF RTS can be divided on 3 groups.

The first group unites RTS, which take parts immediately in technological process. The systems work in real time scale and are controlled with the help of devices with early loaded programs, which operate according to ACS system corrections as a result of diagnostics and measurement systems function. Besides of software there are used also hardware AAS with feedback.

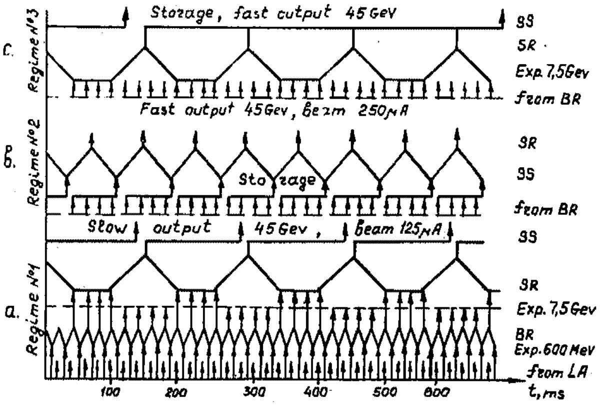


Fig 1. Time regimes of Kaon complex Work

The second group unites measurement and diagnostics systems generally.

The third group includes ACS equipment.

The special features of RTS control depends on two KF properties:

1) the complex consists of cascade row with which of cascade been working in different regimes with different time changing characteristics;

2) the accelerators are fast recycling and high intensity beam devices.

High accelerating process velocity demands increasing of RTS measurement and final-control devices fast-responsibility, localizing of control systems near by the first group RTS and using of distributed control systems. These demands are complicated also by high beam intensity: it needs of accelerators resonator beam loading compensation and coherent betatron particles oscillations suppressing.

3. KF common synchronization system

The system is designed for pulse generation and pulse distribution for LA, BR, SR and SS work synchronization with locking to LA cycle, to BR and SR injection magnetic fields, as well as to BR accelerating field initial frequency and phase. ACS couplings make possible to fulfill corrections at complex work timely. General number of timer pulses is equal 1162 - for BR, 3140 - for SR, 1058 - for SS.

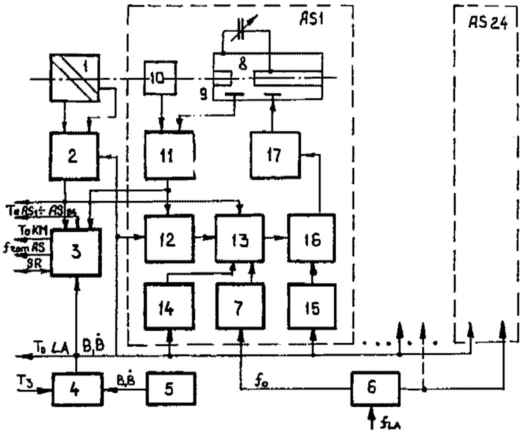


Fig 2. BR frequency control system
1. Beam radial position pick-up. 2. Δr correction signal generator. 3. Cascade-to-cascade beam drive system. 4. B-timer. 5. Primary standard electromagnet. 6. Initial frequency and phase former. 7. Initial frequency phase shifter. 8. Accelerating resonator. 9. Accelerating voltage detector. 10. Beam intensity pick-up. 11. Phase system processing device. 12. $\Delta\Phi$-correction signal generator. 13. Defining generator. 14. Frequency programmer. 15. Amplitude modulation programmer. 16. Amplitude modulator-manipulator. 17. Resonator HF supply system.
f_{LA}-Signal with LA accelerating voltage frequency, $B, \dot{B}$-signals, that are proportional to magnetic field value and its derivative, A_S-accelerating station, T_3-start pulse, KM- kick magnets, T_{0LA}-start pulse LA.

4. KF Accelerating Stations (AS) frequency control system

The system structure for BR is shown at Fig.2. Frequency control complex supplies generation of signals with frequency, phase and amplitude adjusting in accordance to given lows. The signals are used as input AS signals. BR frequency control system concerns with LA frequency, because of "zero"-frequency at injection time must be determined by LA beam bunch following frequency. Block 6 transforms 198.2 MHz signal to 30.03 MHz signal. The last is **used** both for synchronization of 13 generators **in each** of 24 BR AS, and for control of chopper, which **cuts out** of LA beam each sixth bunch. It makes possible to **optimize** putting of injected from LA bunches in BR separatrix along phases and momentums. Before injection beginning controlling generator 13 output signals are directed by modulator-manipulator 15 to AS HF amplifiers 17 inputs. After injection synchro-pulse is switched off and further up to beam injection time in SR work of BR is fulfilled in standard way under programmed frequency control, which is concerned with magnetic field and with frequency correction on base of data about beam radius and phase (there is use beam feedback).

SR and SS frequency control principles are near to that has been described for BR.

Common each cascade (BR, SR, SS) feedback loop with using information about beam radial position (blocks 1 and 2) and partial feedback loops with using data of beam phase position (blocks 10 and 11) are used for slow synchrotrons oscillations suppressing.

5. Accelerating Stations (AS) Automatic Adjusting Systems (AAS)

Salient features of AS AAS and ACS concern with high velocity of accelerating process, large beam loading (the beam taken off power is 4-5 times above resonator loss power), very high RF power levels, which are required for KF cascades work [2].

So, for KF work supporting it needs to supply SR accelerating resonators with 16.5 MW of summary average power (64 AS with 400 kW of maximum power per unit). At the same time AAS must have increased speed of response and large adjusting range.

Common (for SR, BR and SS) AS structure is presented at Fig.3. It consists of some AAS in combination with programmed devices. They support beam and RF accelerating voltage synchronism and maintain longitudinal beam stability. For this reason AS resonator RF voltage is adjusted by AAS of amplitude (AAA) and phase (AAP), as well as natural resonator frequency is adjusted by AFA with fast acting varactor tuner.[1] To maintain stable interaction of all

---

1) It is discussed now opportunities to use varactor tuner for high intensity accelerated beam instability suppressing.

the systems in condition of large beam loading a large amount of RF feedback applied to RF generator and resonator. For transient beam loading compensation during bunched beam injection and because of empty separatrisses (for kick magnets switching) it is suggested to use broadband adjusting loop with beam envelope feedback and one turn delay.

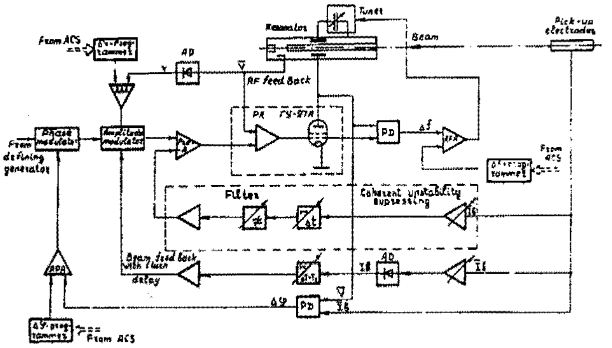


Fig 3. Structure of Accelerating Station with high speed Automatic Adjusting Systems
PA-power amplifier; Pr-A-prime amplifier; PD-phase detector;
AD-amplitude detector; I_B - beam current; V- accelerating voltage.

6. KF RTS Automatic Control System (ACS)

It is suggested to build ACS in such way, that each of KF cascade will have an opportunity to work both in autonomy, and in complex. ACS divided on sectors (clusters) by territorial (for distributed RTS) and functional (for localized RTS) attributes. Cluster can contain separate high productivity microcomputers or computer processor stations of bus-module system crates. Number of connected with ACS information and final-control points is evaluated as ~ 30000. KF ACS structure is shown at Fig.4.

The upper system level furnishes: user access to processors and complex equipment, fulfilling of difficult calculations, that connect with process modeling, work regime resetting of complex on the whole, refusal and failure situations prediction; data base management for complex on the whole; overlapping of current tasks solving with soft- and hard-ware development. The level will be created on base of 2 super-mini computers, for example VAX. These computers are connected to ring network as well, as computers for KF cascade control maintaining. Cluster computers of BR, SR, SS are based on microcomputers, which are connected in local ring network of each cascade.

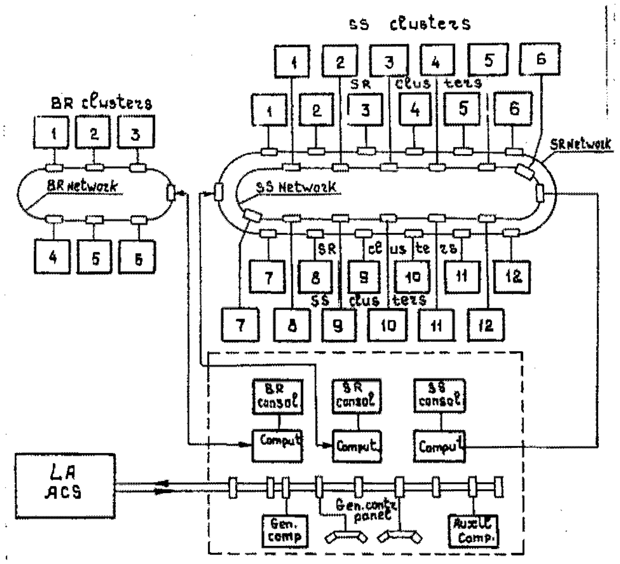


Fig 4. Kaon facility ACS structure

7. Linear Accelerator ACS

LA ACS is built in another way then other KF cascade ACS, because of it has been designed at well earlier time and now it is using already during setting up work on meson facility LA. The system is designed to have 2 hierarchical levels radial network (Fig.5.) [3]. The first (upper) level represents 5 computers of CM-1420 type with standard hardware, coupling and synchronization box equipment and central control console (CCC). By their function keys the computers may be divide on central (the main control computer complex), CCC servicing, library keeping and storing, program debugging and network front-ending (for mutual communication between separate LA ACS computers). The second (cluster's) level is divided by territorial attribute on 5 sectors, which cover LA injector (sector1), LA initial part (sector2) and LA main part (sectors 3-5). Each sector control computer complex is based on CM-1420 computer type also. These complexes are connected with technological equipment through Object Coupling Devices (OCD). As a rule, each of OCD operates by corresponding part of LA technological system (for example, one resonator with RF amplifier channel and

AAA, APA, AFA).

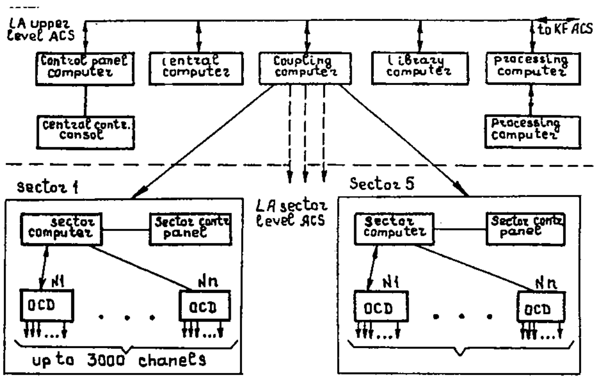


Fig 5. Linear Accelerator ACS structure
OCD - object coupling device

It should be noticed, that some automated LA work control tasks - such, as technological equipment turn-on/-off and bringing to given work conditions, synchronizing processes and setting procedures, - are fulfilled by ACS equipment on real time scale.

8. Conclusion

In conclusion it should be noticed, that up to now there have been worked out the main RTS devices and defined ACS structure and components. In MRTI there are developing full scale brass board models of AS for BR and SR, including ACS and AAS equipment.

Introduction in AS components of new fast-acting varactor tuner, which is able to control high reactive power flows, gives presumptions to use the tuner not only for resonator auto tuning, but and for high intensity beam instability depressing.

9. References

1. Голубев Н.И., Есин С.К. и др. Ускорительный комплекс Каонной фабрики. Всесоюзное Совещание по ускорителям заряженных частиц, октябрь 1988 г., Москва.
2. Иванов Ю.С., Коновалов В.А., Мурин Б.П. и др Особенности разработки ускоряющих станций Каонной фабрики ИЯИ АН СССР, там же.
3. Линейные ускорители ионов. том 2, Основные системы (под редакцией Б.П.Мурина), М., Атомиздат, 1978, с.272-286.

The Development of RF Reference Lines and a Timing System for Japan Linear Collider

J.Urakawa, T.Kawamoto, Y.Otake, Y.Satoh,
T.Mimashi and S.Araki
National Laboratory for High Energy Physics(KEK),
Oho 1-1, Tsukuba-shi, Ibaraki-ken, 305 Japan

Abstract

The main linac of Japan Linear Collider(JLC) will be operated at an X-band frequency of 11.424GHz. The positioning of the X-band accelerating structures at JLC requires precise phase synchronisation over about 10km. Temperature compensated fiber optic cables will be used for the transmission of the 11.424GHz RF signal. The performance of this transmission line is described. Many timing signals will be also transmitted from the main control room, in which the master RF frequency generator will be situated, via this 1.3μm single mode fiber optic link. The outline of the timing system for JLC is given in this paper.

I. INTRODUCTION

A. General

Japan Linear Collider(JLC) is a future project and an electron-positron collider for the energy frontier physics in TeV region. In order to realize the JLC project, we have been discussing for several years on possible parameter sets of the JLC. Fig.1 shows the layout of the JLC according to the parameter set so far obtained[1,2,3].

The main beam parameters of the JLC are shown in Table 1. One of the characteristics of the present design of the JLC is to operate in a multi-bunch mode. The linac accelerates bunch trains where the bunches contained in a train are separated by about 42cm(1.4nsec) and the number of particles per bunch is 2×10^{10}. Fig.2 shows the bunch structure of the JLC.

Table 1 Design Parameters of the JLC

Center of Mass Energy	E[TeV]	0.5	1	1.5
Luminosity	L[$cm^{-2}sec^{-1}$]	2.2×10^{33}	8.8×10^{33}	1.3×10^{34}
Total Length of JLC	L[km]	25	25	25
RF Frequency	f_{rf}[GHz]	11.424	11.424	11.424
Accelerating Gradient	G_a[MeV/m]	40	80	120
Repetition Frequency	f_{rep}[Hz]	150	150	150
Particles/Bunch	N	1.3×10^{10}	2.0×10^{10}	2.7×10^{10}
Bunches/RF Pulse	N_b	20	20	20
Wall Plug Power	P_{AC}[MW]	30	120	240
Average Beam Power	P_b[MW]	3.0	9.7	19.3
Horizontal Normalized Emittance	ϵ_{nx}[radm]	5×10^{-6}	5×10^{-6}	5×10^{-6}
Vertical Normalized Emittance	ϵ_{ny}[radm]	5×10^{-8}	5×10^{-8}	5×10^{-8}
Beam Size at IP	σ_y/σ_x[nm]	4.6/335	3.2/372	2.9/560
R.M.S. Bunch Length	σ_z[μm]	152	112	95
Energy Loss by Beamstrahlung	$\Delta E/E$[%]	5.1	15	15
Circumference of Pre-DR	L_{PDR}[m]	60.1	60.1	60.1
Circumference of Main-DR	L_{MDR}[m]	163.3	163.3	163.3

The JLC timing system is divided into fast and slow timing systems. Fast timing signal transmission system must achieve the timing accuracy within 1$psec$ over the temperature range from 23 to 27°C and over 12.5km from the main control room. For the precise timing signal transmission, a optical fiber cable was developed[4]. This fiber cable showed the reduced thermal transmission delay change less than 10psec/km in the temperature range from -20 to 30°C(average 0.04ppm/°C), which is 100 times smaller than that of any other existing coaxial cables and conventional optical

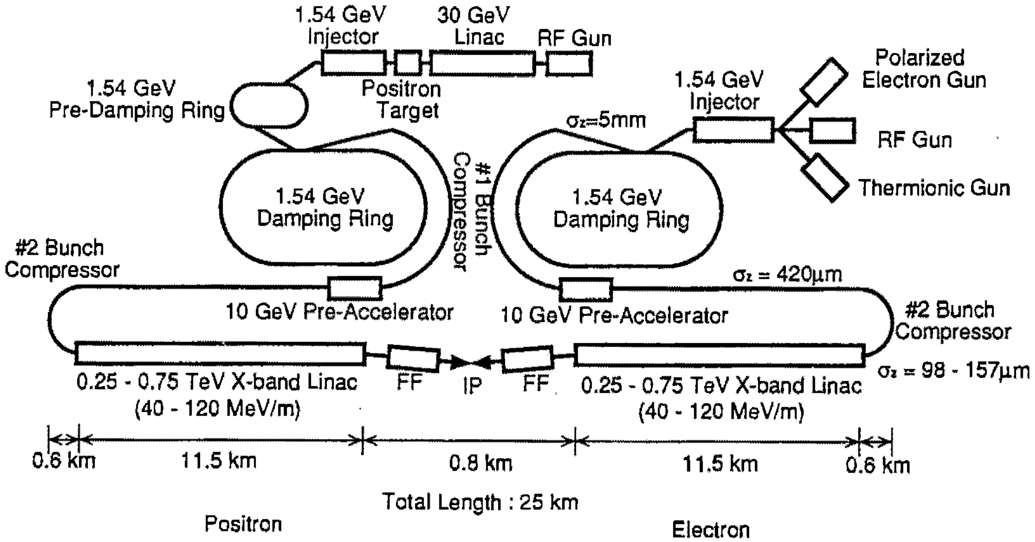


Fig. 1 Layout of the JLC

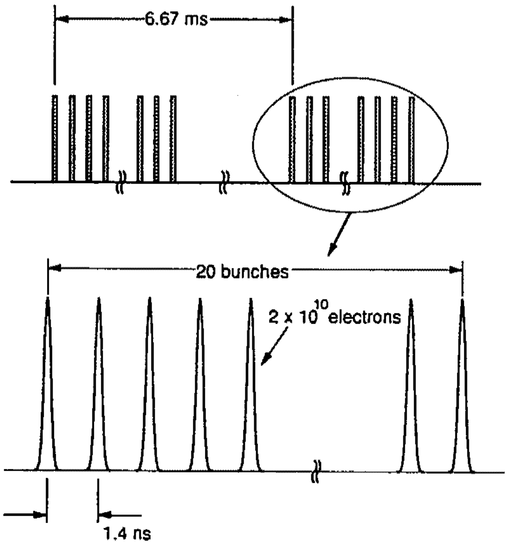


Fig.2 Beam Structure of the JLC

fiber cables. The new optical fiber cable will be installed on a cable rack in the tunnel, in which many klystrons and modulators will be set. In the slow timing system, a dedicated microprocessor for the generation of slow timing signals will be introduced in the computer network system for the JLC control[5]. The slow timing signals will be also transmitted through other fiber in the cable which contains many fibers in it. This communication line is necessary for changing the operation mode and for monitoring synchronized with the beam pulse on a pulse-to-pulse basis.

The fast timing system supplies timing signals(fast timing) for devices whose operation is synchronized with bunched beams. These signals are also used in various beam monitors and beam feedback systems. The slow timing system generates trigger signals(slow timing) in order to achieve synchronization between the beam and the computer processing. These triggers are also used for the automatic operation of machines. The slow timing system manages the operation mode of machines with both flexibility and extensibility. The synchronization signals are transmitted through optical fiber cables over 12.5km from the main control room.

In this paper we describe results and status of the research and development for the RF reference lines. Also, the conceptual design of the fast timing system is described since the outline of the slow timing system was shown in other paper[5,6].

B. Requirements

The beam control and instrumentation of the JLC requires very high precise RF reference lines and timing system. Following requirements are mainly thought to be necessary.

1. The jitter requirements for the gun trigger is determined by the stability of number of particles per bunch (± 0.5%) and the energy acceptance of the electron damping ring (± 0.7%). We must reduce the beam timing jitter to less than ~5psec (rms) with the enough accuracy of electron gun trigger.

2. The phase stability of the RF system for the rings must be reduced to less than ± 0.1 degree (rms) because this value determines collision point in the interaction region.

3. The energy spread of the beam in the main linac must be decreased to less than ± 0.1% (rms) because the energy acceptance in final focus system is small. Then, the signal accuracy of X-band reference line must be reduced to less than ~ 1psec (rms). We maybe also introduce two kinds of structures with a little different accelerating frequency as the tool for the compensation of energy gain variation due to beam loading[7].

4. Since beam instrumentation and control on a bunch-to-bunch basis or on a pulse-to-pulse (train-to-train) basis are essentially required, we must make precise beam timing signals at any local place over 12.5km from the main control room.

II. RESULTS OF THE RESEARCH AND DEVELOPMENT FOR THE RF REFERENCE LINES

A. The Optical Fiber Cable

The thermal coefficient for transmission delay is negligible around 20°C. This optical fiber has the following characteristics:

(1) Stable transmission delay time (less than 0.04ppm/°C),

(2) Low loss (0.35dB/km),

(3) Immunity to electro-magnetic interference,

(4) High resistance against radiation.

Even in the whole expected operation range -20 to +30°C, the transmission delay change is only within 10ps/km. These values are far better than any other existing cables. Fig.3 shows the measured transmission delay time change against temperature for the fiber cable. For the comparison, typical data for the conventional fiber is also shown in Fig.3[8].

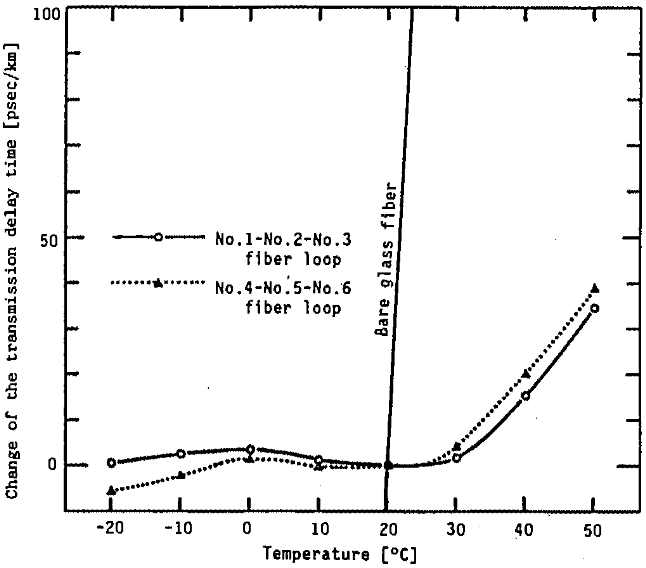


Fig.3 Temperature dependence of the transmission delay time

This fiber cable was installed from the gun room of LINAC to the main control room of the TRISTAN AR in April 1989. The cable has 800m length and contains 6 fibers in it. 300 meter of the total cable was laid even in the underground Positron Beam Transfer Line where the cable was subjected to the irradiation. Since the core material of this fiber is pure silica, this fiber cable is much resistant to irradiation than conventional fiber cables. The characteristics has not been degraded during the last two years.

The confirmed high stability suggests that the use of this cable system will effectively simplify and improve the "main drive lines" in the acceleration systems, where large diameter coaxial cables are used in the special conduit with sophisticated temperature control.

B. Test Results

For the further characterization of the system, the two experiments were conducted. Two other fibers in the cable were connected together at the gun room so that they made a 1600m link with both input and output ends locating at the control room. Using this 1600m link with measuring setup shown in Fig.4 and Fig.5, jitter of the 508MHz signal and drift of the 11.424GHz signal were evaluated.

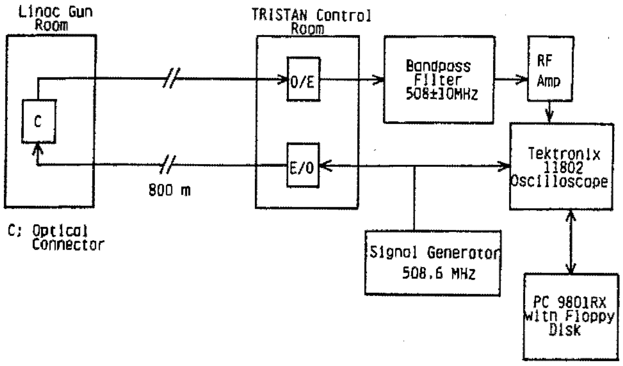


Fig.4 The Circuit for the Measurement of the Timing Accuracy

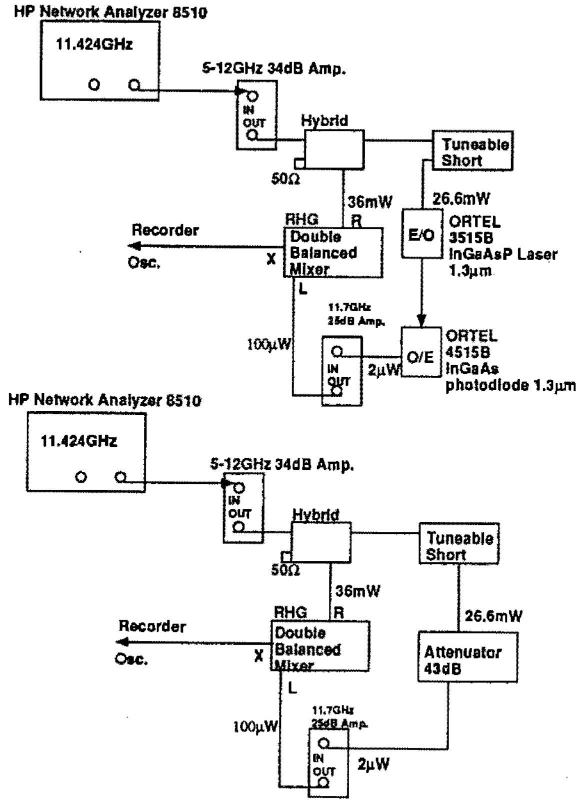


Fig.5 The Circuit for the Measurement of the Phase Accuracy

Phase jitter was measured by a Tektronix 11802 Oscilloscope, and recorded every 40 seconds for 24 hours to evaluate the long-term drift. The long-term drift over the 1600m link transmission was almost negligiblly small(Fig.6). Detail report of this jitter measurement was given in other paper[8].

Fig.7 shows the drift of the 11.424GHz signal with fiberoptic devices from Ortel Corporation and without them. The temperature in the experimental room changed in the range of 23.1 to 25.5°C due to air conditioner. 4 degrees correspond to the phase drift of 1.0psec for 11.424GHz RF signal. Since the double balanced mixer of RHG can respond until 1GHz, we observed that the phase jitter within the bandwidth of 1GHz was almost small. The complete measurement of the phase jitter is in progress.

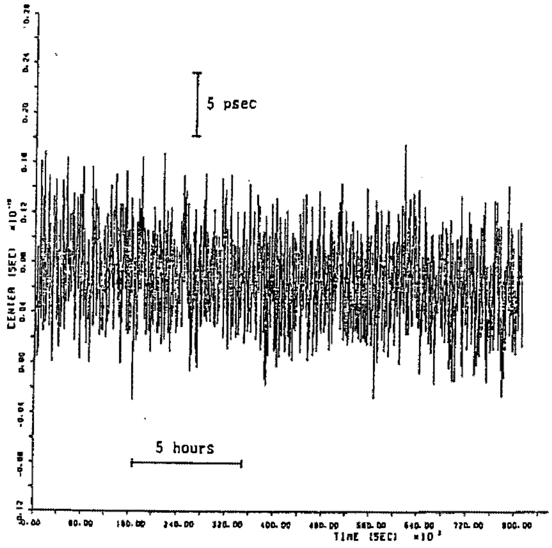


Fig.6 Jitter Fluctuation over 24 Hours

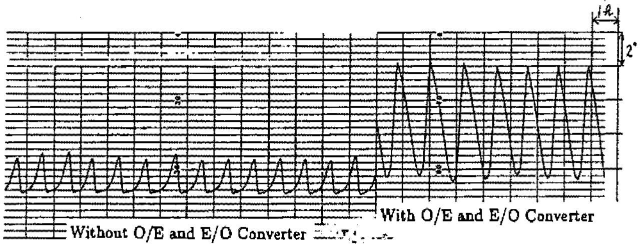


Fig.7 Phase Drift of the 11.424GHz Signal. Horizontal scale; 1hour/div., Vertical scale; 2 degrees/div., 2 degrees correspond to about 0.5psec.

III. CONCEPTUAL DESIGN OF THE FAST TIMING SYSTEM

Many devices for linear collider research and development require precise timing signals. The fast timing system provides timing signals for the pulsed operation of the gun, bunchers, klystron modulators and other equipment. Depending upon the operation mode, several parameters in the timing system must be controlled. A line synchronization generator in the fast timing system is designed to provide triggers(150Hz) at a fixed phase of the three-phase AC line frequency to reduce power line AC effects in accelerator operation. The timing accuracy of this generator is better than ~10nsec(rms). The 150Hz zero crossing signals are synchronized with the 714MHz by a beam timing delay module.

At first, the synchronizer generates a pre-trigger pulse for the S-band linac, the damping ring and the X-band linac. Then the main delay trigger module generates the source of the gun trigger pulse by delaying the pre-trigger. Fig.8 shows the conceptual

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S10TS04

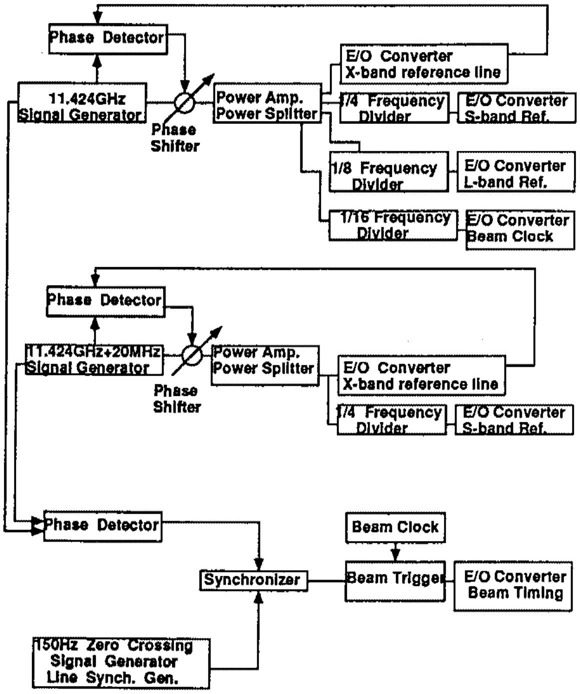


Fig.8 Conceptual Design of the Timing System for JLC

design for timing system which is intended to develop as RF reference lines and beam timing transmission lines for the JLC.

IV. STATUS OF THE RESEARCH AND DEVELOPMENT

The KEK Accelerator Test Facility(KEK-ATF) for the JLC is in construction. The ATF consists of the following major accelerator components; 1.54GeV S-band injector, damping ring, bunch compressor, final focus test facility, 0.5GeV X-band linac and a test station for positron production[9]. Since the S-band injector linac will be completed until March of 1993, following low level modules are being developed for the timing signal transmission system by using GaAs integrated circuits from Nippon Electric Corporation(NEC).

1. μPG506B;11.424GHz 1/8 dynamic prescaler
2. μPG501B;2.856GHz 1/4 dynamic prescaler
3. μPG502B;2.856GHz 1/2 dynamic prescaler
4. μPB587G;0.05GHz-1GHz 1/2, 1/4, 1/8 prescaler
5. Low level power amplifiers
6. Phase shifter
7. Fast phase switch, etc.

In order to investigate the characteristic of materials for RF circuit board, we are designing above frequency divider circuits on three kinds of board. We will decide the material until April of 1992 and complete the timing system for the S-band injector until end of 1992.

We have transmitted 11.424GHz reference signal over 1.6km and obtained enough phase stability. The measurement of the jitter was only shown in the case of 508.6MHz transmission. We are planning to measure the jitter in the case of 11.424GHz transmission. The developments for low level control circuits, especially RF frequency dividers, are in progress.

V. ACKNOWLEDGEMENTS

The authors would like to thank Professors Y.Kimura, Shigeru Takeda and Seishi Takeda for their support during the work. They also wish to thank Dr. S.Tanaka and Mr. Y.Murakami of SUMITOMO Electric Industries for the collaboration in the construction and the test of the optical fiber transmission system.

VI. REFERENCES

[1] Y.Kimura, Proc. of the 2nd European Particle Accelerator Conference, Nice, France, June 12-16, 1990, pp.23.

[2] K.Takata, Proc. of 1990 Linear Accelerator Conference, Albuquerque, New Mexico, Sep. 10-14, 1990, pp.13.

[3] S.Takeda, Particle Accelerators, Vol.30 (1990) pp.143.

[4] J.Urakawa, Particle Accelerators, Vol.29 (1990) pp.251.

[5] J.Urakawa et al., Nucl. Instr. and Meth. A293 (1990) pp.206.

[6] J.Urakawa et al., Nucl. Instr. and Meth. A293 (1990) pp.23.

[7] T.Higo, Private Communication

[8] S.Tanaka et al., KEK Report 90-5 (1990)

[9] Proceedings of the Second Workshop on Japan Linear Collider(JLC), KEK, 1990, KEK Proceedings 91-10(1991)

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S10TS05

A NEW VME TIMING MODULE: TG8

C.G. Beetham, G. Daems, J. Lewis, B. Puccio

European Organization for Nuclear Research (CERN)

CH - 1211 Geneva 23, Switzerland

Abstract

The two accelerator divisions of CERN, namely PS and SL, are defining a new common control system based on PC, VME and Workstations. This has provided an opportunity to review both central timing systems and to come up with common solutions. The result was, amongst others, the design of a unique timing module, called TG8.

The TG8 is a multipurpose VME module, which receives messages distributed over a timing network. These messages include timing information, clock plus calendar and telegrams instructing the CERN accelerators on the characteristics of the next beam to be produced.

The TG8 compares incoming messages with up to 256 programmed actions. An action consists of two parts, a trigger which matches an incoming message and what to do when the match occurs. The latter part may optionally create an output pulse on one of the eight output channels and/or a bus interrupt, both with programmable delay and telegram conditioning.

I. INTRODUCTION

Until the advent of the Large Electron Positron (LEP) collider each new accelerator built at CERN had its dedicated timing system. Each system was tailored to the specific needs of the machine and integrated into the control system. As the working life of a large accelerator spans several decades, it becomes necessary after a certain period to update the control and timing systems. Such an upgrade was applied to the SPS timing system in 1985 [1] and subsequently it was decided to adopt the same system for LEP.

The rejuvenation of the PS control system is presently being implemented [2]. The TG8 timing module, described in this paper, will form an integral part of the joint PS/SPS/LEP timing system. The TG8 is based on a similar but simpler module, the TG3, currently in use at the SPS and LEP.

II. ACCELERATOR TIMING SYSTEMS

The typical real time response of a large accelerator control system is in the region of 10 to >1000ms. Whilst this is adequate for many applications there always remains the requirement to activate equipment with a finer real time resolution. Such an application would be the ramping of the main power converters around LEP. This is achieved by using a separate timing system.

General Machine Timing (GMT) systems for large accelerators normally consist of three parts:

- a central timing generator
- a distribution network
- receiving modules

This article concentrates on the VME receiving modules, although the other two points are also discussed.

The central timing generator is constructed on a single IBM/PC compatible card and is referred to as the Master Timing Generator (MTG). It is basically a large memory which is pre-loaded with machine related timing information for each cycle. At specific times the MTG broadcasts this information over the distribution system. The MTGs for each machine are synchronized to a CERN wide 1ms reference clock.

For the SPS and LEP machines it was decided to use Time Division Multiplex (TDM) techniques, conforming to CCITT Recommendations, to form a backbone for the distribution network. This was adopted due to the long distances involved and also to reduce the number of cables required in the two tunnels. Because of its much smaller size the PS will retain the use of dedicated cables. The overall timing transmission standards used at CERN have been described elsewhere [3].

The receiver modules (TG8s), so named because they have eight output channels, are connected to the timing network and receive information in the form of frames referred to as events. The use and operation of the TG8 is described in more detail later.

III. EVENTS

A. *Standard events*

Standard events mark precise times within a machine cycle, i.e transition, start fast extraction, end of flat top etc. In particular, they are used to initiate actions in the TG8 action table. Such an event has a "header", identifying it with a given machine, plus a one byte code specifying which standard event it is. In addition to this, each standard event is also tagged by a cycle type, and the occurrence number of that type in the super cycle, namely the Cycle Number.

B. *1kHz Clock Events*

The 1kHz event is used to synchronize up to seven preceding events which may have arrived during the previous millisecond period. This type of timing frame is thus specially treated by the TG8 hardware, and in addition to its synchronization usage, it may also be used to clock any of the eight TG8 counters. Like any other event, the 1kHz events can be used to trigger TG8 actions.

C. Date and Time of Day Events

The MTGs receive the Central European Time information each second via a 77.5kHz radio signal originating from a 50 kW transmitter located at Mainflingen, Germany. They receive this information in the form of an ASCII string and subsequently convert the data into eight bytes containing the BCD values for second, minute, hour, day, month and year. The MTG broadcasts these events precisely over the network each second. They have lower priority than the other events so that if a clash occurs then for example, the year or month events will be transmitted during the next millisecond time slot.

In addition, a time event is generated containing the second, minute and hour information and also a date event comprising of the day, month and year. These two events can be used to trigger actions at specific calendar or date times.

D. Telegrams

The telegram message is composed of a set of numbered parameters which describe the particle beam, currently being produced by a group of machines. For historic reasons, these parameters are referred to as groups, a term arising from the way the telegram used to be distributed as a serial bit stream. Each parameter is described by two quantities, namely; the group number, and the corresponding group value. The group number determines which parameter is selected, and the group value specifies a signed sixteen bit value belonging to the selected parameter [4]. Telegrams are associated with a machine, or group of machines, in the same way that standard events are. The telegram message however, do not trigger an action, rather they are used to condition whether or not an action starts executing in the first place.

IV. TIMING FRAMES

The network distributes information as a sequence of four byte frames; up to eight of these timing frames can occur each millisecond. The present SPS/LEP timing system generates four frames per millisecond which has so far proved to be adequate. A timing frame has an identifying header byte, which classifies what kind of information it is (telegram, event, time of day etc.) followed by three bytes which further specify the frame within the header class. The timing frames are transmitted during the one millisecond time slots and are validated at the end of this time. Thus the resolution of a frame, and hence any timing event on the network is one millisecond.

The timing slot boundaries are in fact marked by the 1kHz clock frames, leaving seven time divisions in which other frames can be sent. Thus no more than seven timing events can occur per millisecond, as the eighth division is used by the millisecond clock.

V. SIGNAL STANDARDS

The original machine timing signal standard used at CERN was developed over 30 years ago. This "blocking oscillator" circuit generates a transformer isolated pulse with an amplitude of 24 volts and a duration of 1.5us. It was designed specifically to transmit timing pulses over 50 Ohm co-axial cables in hostile and noisy environments. It is still in use throughout the PS and also in the original SPS control system.

However, for the SPS upgrade and also for LEP, it was decided to adopt commercial standards rather than continue with an in-house system. The standard chosen conforms to the electrical characteristics defined by CCITT Recommendations V.11 and X.27 and EIA specification RS-485. Each frame is Manchester encoded and the local line drivers are transformer isolated at the driving end. The data format adopted is supported by integrated circuits from National Semiconductor (the NS8342/8343 transmitter/receiver set).

VI. TG8 HARDWARE

A. Overview

The TG8 is a timing module conforming to the VME standard. It operates in the slave mode and also as an interrupt generator. The TG8 VME timing module consists of two sections: a receiver part and a process part. The basic purpose of the receiver part is to accept all the frames from the timing network and to pass on the treated events to the process part. The process part compares each received event with a set of pre-loaded parameters contained in a portion of the on-board memory referred to as the "action table". These parameters are a subset of the total events contained in the MTG's memory.

If a valid comparison is found between the received event and the parameters loaded in the "action table", then that specified action is performed. This will normally result in interrupting the VME crate's CPU, and/or generating a TTL level pulse on one of the module's eight front panel outputs. The options are user programmable.

B. The Receiver Part

In the MTG, the NS8342 integrated circuit frames the 32 bit events into four bytes. Each frame is enveloped within a predefined start/end sequence and in addition each byte starts with a synchronizing bit and ends with a parity bit. The TG8's NS8343 receiver chip performs the reverse operation and reconverts the frame to a 32 bit NRZ word. In addition, the NS8343 contains a seven bit error register which indicates, amongst other things, the detection of a mid-bit transition fault, an invalid ending sequence and also a parity error.

The contents of this register are used by the TG8 for error detection.

The receiver continuously monitors the reception of the 1ms clock events. If a clock is missing then the hardware generates a substitute clock, produces a 1ms "watch-dog" error and informs the process part of the error situation. All of these error bits can be read via the VME bus.

The interface between the NS8343 integrated circuit and the process part incorporates a XILINX 3030 gate array. It contains all the logic necessary to perform the above functions.

C. *Process Part*

The core of the process part is the Motorola MC68332 integrated circuit. This chip is a powerful microcontroller based on the MC68020 microprocessor. It contains the various functions required for embedded control applications such as 16 bit timers, RAM, UART, along with digital input and output facilities. A second XILINX is used, this time a 4008 containing 8000 gates, which contains all the logic to control the counters and interrupt circuits for each of the eight outputs.

The module uses 16 bit VME data transfers and 24 bit address decoding. In order to optimize communications between the VME bus and the timing card, two kbytes of the TG8's total memory are dual ported. These two kbytes can be accessed both from the VME bus and the MC68332 microcontroller. For system configuration reasons, this dual port memory can be placed on any location in the 16Mbyte VME memory map. The module also contains 1Mbit of EPROM, used to contain the microcontroller's firmware, plus 512kbits of static RAM along with a 256*48 bit Content Addressable Memory (CAM) which contain various user tables.

The firmware's main task is to treat interrupts generated by the receiver part. For every valid frame received the MC68332 uses the speed of the CAM to compare each event with up to 256 pre-loaded trigger conditions. If a valid comparison is detected then the microcontroller, using the XILINX gate array, loads the specified counter with the appropriate delay, selects the clock unit and prepares the hardware to generate an output pulse and/or a VME interrupt when the selected counter has been decremented to zero.

D. *The Action Table*

The action table is partitioned into two sections referred to as the trigger part and the action part. The trigger part is located in the CAM, whilst the action part is located within the RAM.

The trigger part contains the full 32 bits required for the event definition, plus the 16 bit PLS identifier. The action part describes what the TG8 must do when triggered, such as setting up a counter to make an output pulse etc. With the exception of the first byte header, "don't care" values (Hex.FF) can be placed in any of the other three byte fields. In this case they will always match with the corresponding field of the event being processed, no matter what it contains. Thus up to three "don't care" wild cards can be used in order to specify a trigger condition.

VII. TG8 SOFTWARE

From a software point of view, the major task is the management of the action table. As previously mentioned, the table contains 256 rows each of which describes a unit of work to be carried out by a TG8. Each row is composed of a trigger and an action to be performed when the trigger conditions occur. A TG8 unit of work is specified by:

1) The trigger condition, which is composed of an event specification and an optional telegram condition. Both of these must occur before the rest of the action table row is processed.

2) The delay, which consists of a clocking train selection, and the number of ticks to be counted in the specified train.

3) The mode, which describes how the TG8 counters may be combined to produce counter chains, in which the output of one counter triggers the start of another; or burst mode, in which pairs of counters are combined to produce a burst of pulses.

4) The result part, which will be either a VME bus interrupt, or an output pulse, or both. If a bus interrupt is specified, then a user specified callback routine will be invoked. A library for the VME chassis processor will be provided.

Each TG8 module in a given VME crate is controlled by a device stub controller (DSC) processor [5], which runs a diskless real time POSIX-4 compliant version of Unix supporting all the usual features such as NFS, TCPIP, X-Window client etc. Applications running at this level use the TG8 library to control the TG8 modules resident in the same VME crate, and are able to communicate with Unix based work stations running X-windows and OSF Motif.

The application program interface (API) will consist of a C language binding to a TG8 library in which functions manipulate structures containing action table rows. Care must be taken when combining these rows into an action table for a particular TG8 Checks must be carried out to ensure that the individual rows do not compete for TG8 resources, such as counters, in real time. Some simple checks can be performed which ensure a basic level of consistency, but full checking would depend on knowing how the trigger conditions will actually occur in real time, which is clearly impossible.

VIII. CONCLUSIONS

The consolidation of CERNs different machine timing systems has been proposed and discussed many times during the past two decades, without significant results. This present proposal was initially outlined at a joint PS/SL Control Users

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S10TS05

Forum held in Chamonix, France, during April 1990. In June 1991 the PS and SL control groups agreed on a specification for a common machine timing system [6]. This entailed using the IBM/PC based Master Timing Generator, as presently used in the SL division and to produce a new VME timing module compatible with both timing systems.

Presently, the prototype TG8 module is being tested in the laboratory using the SL timing system. It is planned to test the unit with the PS system in the first quarter of 1992. It is therefore too early to arrive at any definite conclusions, particularly regarding costs. The price of the "state of the art hi-tech" components used in the TG8 is falling rapidly and clearly the production units will be far less expensive than the present prototype. On the cost basis however, one must consider at what stage in a systems lifetime is it cost effective to develop a completely new system rather than to develop complex new modules compatible with two different existing systems.

IX. ACKNOWLEDGEMENTS

Many people have contributed to the upgrading of the PS/SL control and timing systems, but particular mention must be made to the two group leaders, K.-H. Kissler and F. Perriollat, who have stoically pursued the project. Also to P. Nouchi and R. Parker for their invaluable contributions.

X. REFERENCES

[1] C.G. Beetham, R.J Lauckner, C.G. Saltmarsh, Overview of the SPS/LEP Fast Broadcast Message Timing System, Proc. of the 1987 Particle Accelerator Conference, Washington, D.C., Vol. 1, p. 766.

[2] F. Perriollat, C. Serre, Problems in Rejuvenating a Control System and Plans for the CERN Proton Synchrotron (CPS) ICALEPCS Vancouver 1989

[3] C. G. Beetham, K. Kohler, C.R.C.B Parker, J.-B. Ribes, The Transmission of Accelerator Timing Information around CERN, these Proceedings.

[4] J. Boillot, G. Daems, P. Heymans, M. Overington Pulse-To-Pulse Modulation of the Beam Characteristics and Utilization in the CERN Accelerator Complex. Proc. of the 1981 Particle Accelerator Conference, Washington, D.C.

[5] A. Gagnaire, N. de Metz Noblat, Ch. Serre, Cl. H. Sicard, Replacing PS Controls Front End Minicomputers by VME Based 32-bit Processors, these Proceedings.

[6] PS/SL Controls Consolidation Project, Technical Report, CERN PS/91-09(CO), CERN SL/91-12(CO). Unpublished

Modular Pulse Sequencing in a Tokamak System.

A.C.Chew, S.Lee and S.H. Saw
Plasma Research Laboratory
Department of Physics
University of Malaya
59100 Kuala Lumpur
Malaysia

Abstract

Pulse technique applied in the timing and sequencing of the various part of the MUT tokamak system are discussed. The modular architecture of the pulse generating device highlights the versatile application of the simple physical concepts in precise and complicated research experiment.

I. Introduction

In experimental studies of pulse plasma devices, timing and sequencing of the various events are an important part of the experiment and requires careful considerations. This is achieved in the MUT (University of Malaya Tokamak) tokamak system [1] by employing modular architecture involving various modules of pulse generating devices [2].

II. The MUT System

The MUT system consists of the stainless steel toroidal chamber, the toroidal field coil system and the ohmic heating coil system incorporating the vertical field generating design is assembled as shown in Fig. 1. The major radius of the chamber is 25 cm while the minor radius is 5.4 cm. The torus is divided into two halves separated by insulating flanges. The diagram of the top view of the torus is shown in Fig. 2. The vacuum system uses a 300 litres s^{-1} diffusion pump backed by a rotary pump. The system provides a base pressure of 10^{-5} torr. The toroidal field coil consists of 99 turns of insulated copper wire of total cross-sectional area of 0.3 cm^2, coil resistance of 0.02 Ω and inductance of 95 μH, giving a time constant for the toroidal field of 4.7 ms. It is powered by a 4.5 mF, 1.3 kV capacitor bank system while the ohmic heating system are powered by a 5 μF, 20 kV capacitor bank. The block diagram in Fig. 3 shows the sequence of operation of the various stages.

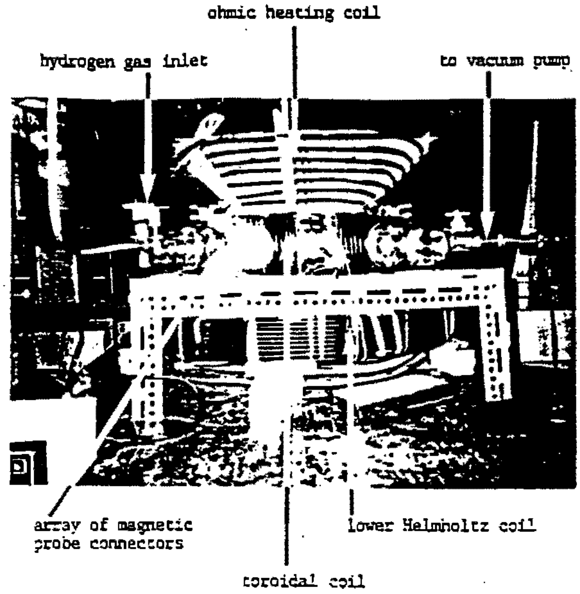


Fig. 1 The MUT system.

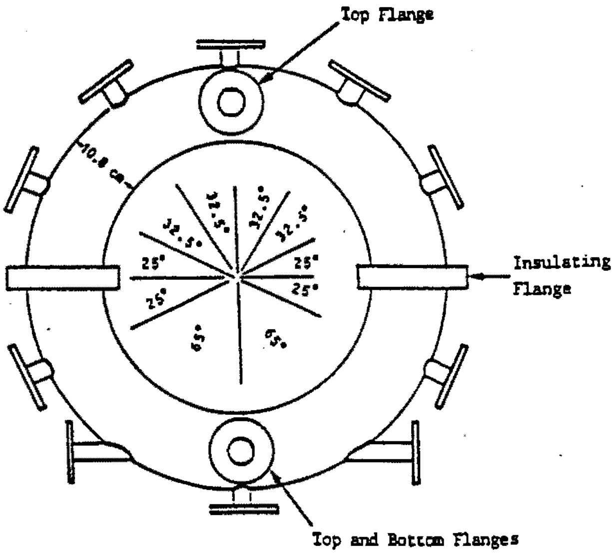


Fig. 2 The tokamak vessel.

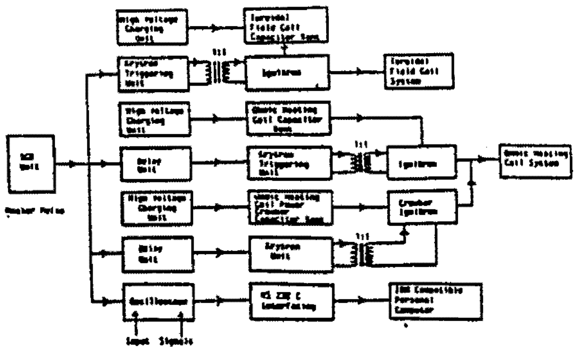


Fig. 3. Block diagram for the operational sequence of the tokamak system.

III. Modular Electronics and Control System.

The electronic system for the MUT tokamak system are specially designed in modular form. The four modules are as described below.

A. Module 1

Module 1 generates the 50 ns master trigger pulse which is produced by a SCR circuit activated manually by a 22.5 V pulse circuit shown in Fig. 4. The 0.1 μF capacitor of the SCR unit generates a pulse when fired across a 50 Ω load resistor. This pulse is sufficient to provide accurate triggering of the units shown in Fig. 3.

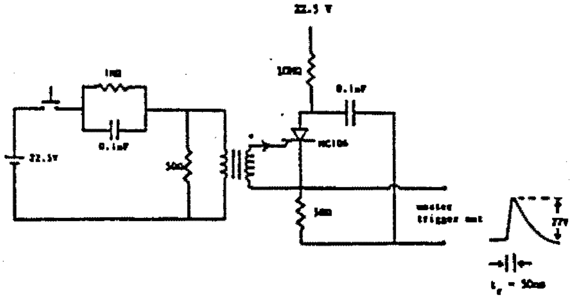


Fig. 4. Module 1 - SCR circuit.

B. Module 2

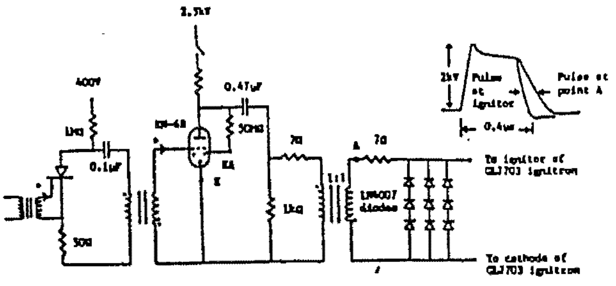
Module 2 consists of a SCR trigger unit. In this module the SCR switches a 400 V, 0.1 μF capacitor upon the introduction of a 20 V, 50 ns pulse from module 1 to the gate via a decoupling trigger transformer . The circuit is shown in Fig. 5(a).

C. Module 3

Module 3 is a krytron triggering unit. It uses a KN-6B krytron to switch a 0.47 μF capacitor charged to 2.5 kV. The pulse circuit is as shown in Fig. 5(b). The krytron is "kept-alive" by a 50 μA current to the KA electrode via a 50 MΩ tapping resistor. The krytron is triggered by the introduction of a pulse from module 2 to the gate of the krytron via a decoupling trigger transformer as before.

D. Module 4

Module 4 generates the ignitor pulse used to switch the ignitron and hence discharge the capacitor energy into the tokamak system. The circuit is shown in Fig.5(c). It consists of a 1:1 decoupling trigger transformer. The secondary of the transformer is connected via a 7 Ω protective series resistor to the ignitor of the GL 7703 ignitron. The ignitron is protected from pulse reversal (not to exceed 25 V) by a bank of 1N4007 diodes. The 1:1 pulse transformer serves to isolate the trigger circuits from the high power capacitors that the ignitron is switching. The krytron circuit in module 4 provides a 2 kV pulse at the ignitor with a current flow of 200 A to create the required hot spot in the ignitron mercury pool necessary for switching on the ignitron. The ionization time (turn-on time) of the GL 7703 ignitron is 0.4 μs.



(a) Module 2 (b) Module 3 (c) Module 4

Fig. 5 Modules 2, 3 and 4.

IV. CONCLUSION

A modular architecture consisting of the pulse generating devices used in the timing and time sequencing of the tokamak system is described. Its compactness and versatility have allowed its applications in other pulse plasma devices such as the focus [3], pinch [4] and electromagnetic shock tube [5] with minimal modifications.

V. REFERENCES

[1] A.C. Chew, "Design and Development of a Small Tokamak", University of Malaya, Kuala Lumpur, Ph.D. dissertation, March 1991.

[2] S.Lee, "Pulsed Electronic Modular Devices", UNESCO/ ASPEN Physics Equipment Programme, IBSN 967-99993-0 0, 1986.

[3] S.Lee, "Laser and Plasma Technology ", World Scientific, 1985, pp.38-46.

[4] S.H. Saw, "Experimental Studies of a Current-stepping Z-pinch", University of Malaya, Kuala Lumpur, Ph.D. Thesis, Dec. 1990.

[5] S.Lee et. al. Laser and Plasma Technology", World Scientific,1985, pp.3-20.

The Timing System of the RFX Nuclear Fusion Experiment

V. Schmidt*, G. Flor, G. Manduchi, I. Piacentini+
Associazione EURATOM - CNR - ENEA, Padova Italy

ABSTRACT

The RFX Nuclear Fusion Experiment [1] in Padova, Italy, employs a distributed system to produce precision trigger signals for the fast control of the experiment and for the experiment-wide synchronization of data acquisition channels. The hardware of the system is based on a set of CAMAC modules. The modules have been integrated into a hardware/software system which provides the following features:

- generation of pre-programmed timing events
- distribution of asynchronous (not pre-programmed) timing events,
- gating of timing event generation by Machine Protection System,
- automatic stop of timing sequence in case of highway damage,
- dual-speed timebase for transient recorders,
- system-wide precision of $\leq 3\ \mu s$, time resolution $\geq 10 \mu s$.

The operation of the timing system is fully integrated into the RFX data acquisition system software. The Timing System Software consists of three layers: the lowest one corresponds directly to the CAMAC modules, the intermediate one provides pseudo-devices which essentially correspond to specific features of the modules (e.g. a dual frequency clock source for transient recorders), the highest level provides system set-up support

The system is fully operational and was first used during the commissioning of the RFX Power Supplies in spring '91.

1. SIGMA

The Timing System is part of the fully computerized system for experiment control, monitoring, and data acquisition known as SIGMA (Sistema di Gestione, Monitoraggio ed Acquisizione Dati) [2]. SIGMA employs two distinct technologies: PLCs and CAMAC/VAX systems. Nine large PLCs, grouped into four subsystems provide slow control and continuous monitoring of the corresponding machine subsystems. Fast data acquisition and the generation of the precision trigger signals and of the fast reference waveforms is implemented by means of CAMAC equipment connected to a central VAXcluster via a fibre optic CAMAC Serial Highway implementation. All components of the system are connected to the same fibre optic Ethernet.

2. TIMING SYSTEM HARDWARE COMPONENTS

2.1. THE TIMING HIGHWAY

The CAMAC modules of the timing system are connected by a single optical fibre Timing Highway which carries the timing events in encoded form, imprinted on a 1 MHz carrier clock. Each timing event is encoded in a 10-bit frame. The fibre and connector are identical to the ones used both for the fibre optic Serial Highway and for the fibre optic Ethernet.

2.2. CAMAC MODULES

The system employs three types of CAMAC modules which provide the following functions:

The *Encoder Module* generates encoded timing events; it can be seen as the input device of the Timing system. Each Encoder can generate a maximum of seven events, of which six are produced by hardware inputs and one by software command. The event inputs are priority encoded. The code associated with each event is pre-loaded via the CAMAC interface.

The *Decoder Module* is the output device of the Timing System. The module can be divided in two sections: the code-recognizer section, which matches encoded events traveling on the Timing Highway to pre-loaded codes held by internal registers, and the counter-timer section. The module contains a crystal oscillator, which can be used as master clock. The module provides a rich set of options and operating modes

The *Timing Event Recorder* module can log up to 512 timing events together with their relative time of registration with reference to a start event (software or hardware defined).

Encoder and Decoder module have been developed for the Tokamak de Varennes [3]. The Timing Event Recorder has been added by RFX. All modules are commercially available [4].

3. HW SYSTEM DESCRIPTION

3.1. TIMING HIGHWAY STRUCTURE

From an operational point of view, a clear distinction has been made between parts of the timing system which are required to be permanently 'on-line' and which are essential to produce a plasma shot and other parts which could be excluded, intentionally or unintentionally, without preventing normal operation. Hence, the timing system has been separated in two sections: *Machine Section* and *Diagnostics Section* (Fig. 1). The Machine Section delivers timing signals to all the essential machine components, i.e. converter units, gas injection control, essential data acquisition equipment. The Diagnostics

---

* under contract from Hahn-Meitner-Institut Berlin GmbH, Berlin , Germany

\+ now with SPIn s.r.l., Milano, Italy

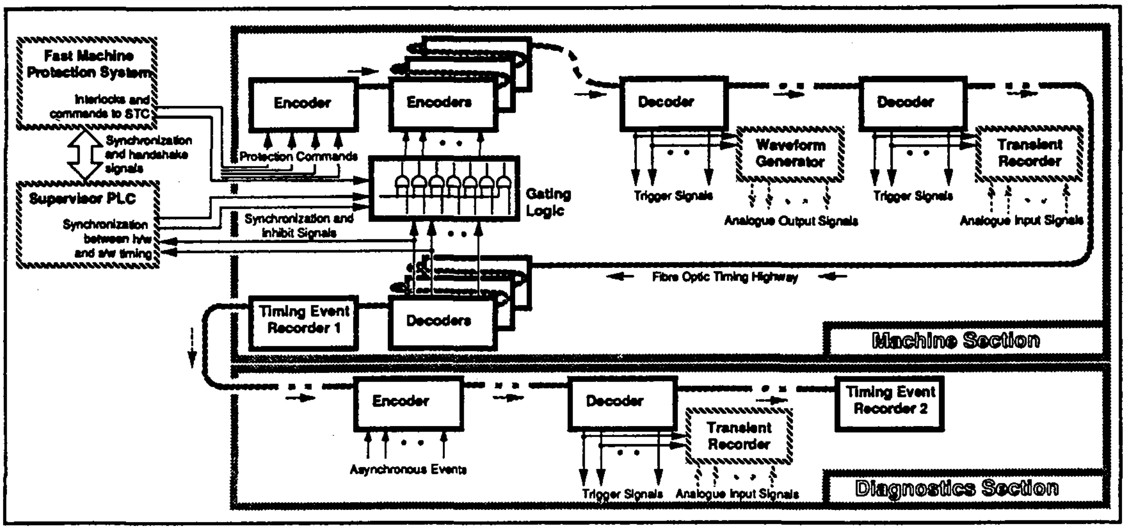


Figure 1: Hardware Structure of the RFX Central Timing System

Section delivers timing signals to all the equipment which is not essential for machine operation, typically the diagnostics. The two sections use one Timing Highway. The highway serves first the Machine Section and afterwards the Diagnostics Section. A timing Event Recorder is placed at the end of each section.

3.2. FEATURES

3.2.1. PRE-PROGRAMMED TIMING EVENTS

A number of pre-programmed timing events are released sequentially after an initial software start command or hardware start pulse. The start signal is fed into an Encoder event input channel. The corresponding event then travels in encoded form along the Timing Highway; it can be used by any downstream Decoder module. It returns to a Decoder module placed at the "end" of the highway (physically near to the Encoder module). This Decoder produces another timing pulse after a pre-programmed delay after receiving the encoded Start event. This pulse is again fed into an event input of the Encoder, from where it travels along the highway as second encoded event. All further timing events are generated in the same way as delayed events after the start signal. This structure has the effect of interrupting the timing sequence in case of interruption of the Timing Highway. In this case the clock signal is missing which is used to count down the delays for the delayed events.

At the local CAMAC crate, one decoder channel is required per event in order to "reproduce" the event as local timing pulse. This reproduction includes a programmable delay, the possibility to invert the output signal, and to program the duration of the local timing pulse.

3.2.2. ASYNCHRONOUS TIMING EVENTS

Encoders which are placed anywhere on the Timing Highway may be used to encode asynchronous external signals. Such events are encoded in exactly the same way as pre-programmed events. They can be decoded and used by any decoder downstream from the position of the encoder module on the highway. At RFX this mechanism is used to record interventions of the independent Fast Machine Protection System and to trigger automatically data acquisition channels in case of such intervention.

3.2.3. GATING OF TIMING EVENTS

Timing events produced by the central part of the timing system are subject to gating by the fast machine protection system. This intervention is part of the machine protection strategy of RFX. The gates are placed at the inputs of the Encoder Modules of the event generation circuits.

3.2.4. TIMEBASE FOR A/D AND D/A MODULES

A large number of A/D Converter CAMAC Modules need to be synchronized during the plasma shot both in terms of sampling repetition rate and acquisition time window. An equivalent requirement exists for the generation of fast, pre-programmed reference waveforms, which are produced by D/A CAMAC modules with local memory. Among the many possible configurations two 'standard' configurations have been selected: The *Gated Single Speed Clock Generator* configuration produces clock pulses of programmable frequency and on/off ratio for a time window which starts at a programmable delay after a selected timing event and ends after a programmable time. This configuration occupies two Decoder channels. The *Dual Speed Clock Generator* configuration produces clock pulses of programmable frequency f1, switches to programmable frequency f2 after a selected timing event, and switches back to frequency f1 after a programmable delay. This configuration occupies two Decoder channels.

3.2.5. PRECISION AND TIME RESOLUTION

The system guarantees an overall precision of less than

3 µs between any two events in any two locations of the system. Time resolution, i.e. the minimum time between two different, programmed or recorded events, is 10 µs which are due to the frame length of the encode timing event on the Timing Highway.

3.2.6. SELF-TEST

Automatic Stop of the Timing Sequence: In case of interruption of the Timing Highway the generation of further timing events downstream from the interruption is inhibited because the internal counters of the downstream decoders are stopped due to the missing highway clock which is used to increment these counters. As the Decoder modules which produce the highway events are placed at the"end" of the highway any such interruption has the effect that no further events are produced.

Event Recording Function: In order to record the time of occurrence of events, the timing system comprises Timing Event Recorder modules which register timing events. The same function is used for recording asynchronous events and for the verification of the generation of the synchronous events.

Missing Clock Detection: The Timing Event Recorder Module which is placed at the end of the Timing Highway can be programmed to produce an interrupt request to the computer in case of missing highway clock.

Watchdog Function: A software controlled watch-dog function can easily be implemented. At regular intervals a specific software generated event is produced and sent down the highway. A Decoder channel placed at the end of the Timing Highway is programmed to detect this event.

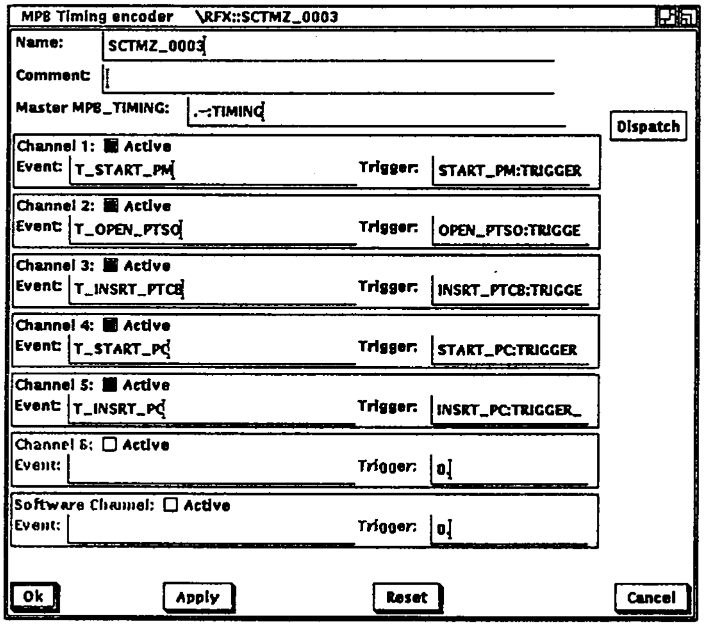
4. SW SYSTEM DESCRIPTION

4.1. MDSPLUS

The Timing System Software is embedded in MDSplus [5], the model driven data acquisition system jointly developed by IGI Padova, MIT Plasma Fusion Laboratory and the Fusion Group of Los Alamos National Laboratory. It works in a VAX/VMS cluster environment. It is based on the concept of an "experiment model" which contains a hierarchical, tree-structured representation of the experiment. The experiment model contains the description of the parameters to be loaded, of the data to be acquired, of the devices used for acquisition, of the set-up of those devices, of the data acquisition and analysis "actions" to be executed during the experiment shot cycle.

4.2 TIMING SYSTEM SOFTWARE

The software support for the timing system has been organized in a three-layer approach: module-related devices, functional pseudo-devices which regroup frequently-used functionalities of modules into easy-to-handle software devices, and one system-wide device.


MPB Timing encoder \RFX::SCTMZ_0003

Name: SCTMZ_0003
Comment:
Master MPB_TIMING: .-:TIMING

Channel 1: ■ Active
Event: T_START_PM Trigger: START_PM:TRIGGER

Channel 2: ■ Active
Event: T_OPEN_PTSO Trigger: OPEN_PTSO:TRIGGE

Channel 3: ■ Active
Event: T_INSRT_PTCB Trigger: INSRT_PTCB:TRIGGE

Channel 4: ■ Active
Event: T_START_PC Trigger: START_PC:TRIGGER

Channel 5: ■ Active
Event: T_INSRT_PC Trigger: INSRT_PC:TRIGGER_

Channel 6: ☐ Active
Event: Trigger: 0

Software Channel: ☐ Active
Event: Trigger: 0

Figure 2: User Software Window for Encoder Module

4.2.1 MODULE-RELATED DEVICES

One such software device exists for each type of CAMAC module, which make all functions and settings of the actual CAMAC modules available: MPB_DECODER, MPB_-ENCODER, and MPB_RECORDER respectively. These are the only software devices which have a direct hardware counterpart. Their fields reflect the hardware registers of the modules. Fig. 2 shows, as an example, the set-up window for the Encoder module.

4.2.2. FUNCTIONAL PSEUDO-DEVICES

The functional pseudo-devices have been provided in order to provide an easier interface for the Decoder module. They do not correspond directly to any hardware. Rather, the contents of their fields are translated by their initialization operation into the appropriate values and stored in the "target" MPB_DECODER device. The pseudo-devices do not support all features of the module, only the more commonly used configurations are supported.

MPB_PULSE provides the software support for the generation of pre-programmed timing events at local CAMAC crates (see 3.2.1 above). It provides the generation of a pulse after a programmable delay from a timing highway event. Optionally a second pulse may be generated on the same output channel after a second delay. Pulses can be substituted with level toggles.

MPB_CLOCK is a continuous single speed clock generator, with programmable frequency and duty cycle.

MPB_GCLOCK provides the software support for the Gated Single Speed Clock Generator (see 3.2.4. above)

MPB_DCLOCK provides the software support for the Dual Speed Clock Generator (see 3.2.4 above). Fig. 3 shows, as an example, the corresponding set-up window.

4.2.3. SYSTEM DEVICE

The system pseudo-device MPB_TIMING has three distinct functions: system initialization, system configuration assistance, highway monitoring.

A typical *initialization* sequence is the following: First, the table of association between timing event codes and the corresponding times (in seconds relative to the start of the experiment pulse) is calculated from the user input to the MPB_ENCODER and MPB_TIMING devices. Next, initialization operations are performed on all declared pseudo-devices by loading the necessary parameters into the MPB_DECODER devices (no CAMAC command is issued in this phase !). If event times are required (e.g. to record the correct pulse time in a MPB_PULSE device), they are retrieved from the MPB_TIMING system-device. Last, initialization operations are performed on the MPB_DECODER and MPB_ENCODER devices and set-up data are loaded into the Decoder and Encoder module registers.

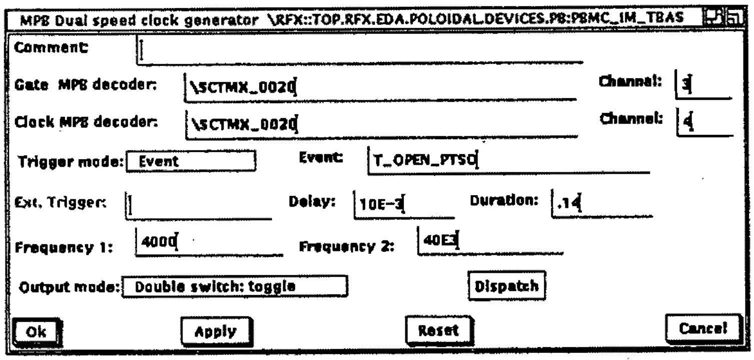
The second function of the system-device MPB_TIMING is the *assistance with system hardware configuration.* In large systems, as RFX, events are generated by an Encoder whose trigger input is the output signal from a Decoder channel which is used as a delayed pulse generator which in turn is using an event produced by another Encoder channel. In addition, gated and dual speed clock generators require two Decoder channels to work: the first channel is used as a gate for the second one. Keeping track of all necessary connections and of the times associated to events and pulses, as well as frequency switches, may become very difficult when a large set of signals is involved. MPB_TIMING supports this

- by calculating the times associated with each event following the cascade of hardware triggers and timing events;
- by providing a graphic display of the set of the programmed signals;
- by listing the required hardware connections.

The third function of MPB_TIMING is the *display of the timing highway events* as recorded by the timing event recorder modules(s). As shown in Fig. 4 the time of arrival of each event is displayed alongside the time at which the event had been programmed.

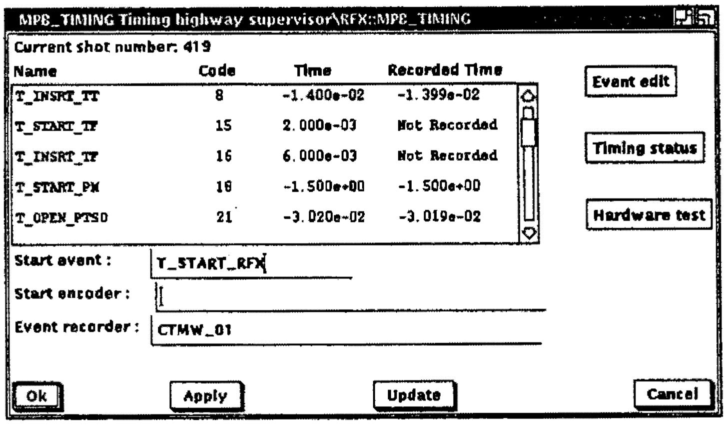
5. OPERATION EXPERIENCE

The system is now fully operational. It was successfully used during the integrated commissioning of the RFX Power Supplies from April to October 1991. It is now being used for the start-up phase of RFX which should lead to first plasma before the end of 1991.


MPB Dual speed clock generator \RFX::TOP.RFX.EDA.POLOIDAL.DEVICES.PB:PBMC_IM_TBAS

Comment:
Gate MPB decoder: \SCTMX_002 Channel: 3
Clock MPB decoder: \SCTMX_002 Channel: 4
Trigger mode: Event Event: T_OPEN_PTSO
Ext. Trigger: Delay: 10E-3 Duration: .14
Frequency 1: 4000 Frequency 2: 40E3
Output mode: Double switch: toggle

Figure 3: User Software Window for Dual Speed Clock Generator Device


MPB_TIMING Timing highway supervisor\RFX::MPB_TIMING

Current shot number: 419

Name	Code	Time	Recorded Time
T_INSRT_TT	8	-1.400e-02	-1.399e-02
T_START_TF	15	2.000e-03	Not Recorded
T_INSRT_TF	16	6.000e-03	Not Recorded
T_START_PM	18	-1.500e+00	-1.500e+00
T_OPEN_PTSO	21	-3.020e-02	-3.019e-02

Start event : T_START_RFX
Start encoder :
Event recorder : CTMW_01

Figure 4: User Software Window for Timing Event Recording

The actual installation comprises of 5 Encoders, 24 Decoders, and 2 Event Recorders. About 20 different timing events are being produced during a normal RFX pulse cycle which are used as trigger signals in different parts of the plant. 5 asynchronous events (from the protection system) are actually recorded and distributed. Time base signals are generated for 52 CAMAC transient recorders providing 391 data acquisition channels, 58 of which use the dual frequency clock feature, and for 10 D/A converter modules which provide 42 fast reference waveforms.

REFERENCES

[1] G. Malesani, G. Rostagni, "The RFX Experiment", in *Proceedings of the 14th Symposium on Fusion Technology*, Sept. 1986, Avignon, France, pp.173-184

[2] V. Schmidt et al., "Structure definition and component selection for the RFX control and data acquisition system", in *Proceedings of the 12th Symposium on Fusion Engineering*, Oct. 1987 Monterey, California, pp. 695-698

[3] J.M. Larsen et al., "The Data Acquisition System of the Tokamak de Varennes", in *Proceedings of the 15th Symposium on Fusion Technology*, Utrecht, The Netherlands, Sept. 1988, pp. 1687-1691

[4] manufacturer: MPB Technologies Inc., 1725 North Service Road / Trans Canada Highway, Dorval, Québec H9P 1J1

[5] G. Flor et al., "MDS-Plus - A Comprehensive Data Acquisition and Analysis System", in *Proceedings of the 16th Symposium on Fusion Technology*, Sept. 1990, London, U.K., pp. 1272-1276

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S10TS08

An Optical Fiber Phase Lock Network of a Radio Interferometer

Masanori Nishio, Susumu Kawashima, Chikayoshi Torii, Hiroshi Nakajima
Nobeyama Radio Observatory, National Astronomical Observatory
Nobayama, Minamisaku, Nagano, 384-13 Japan

Toshio Takabayashi, Katsuhiko Nishikawa, Norio Futagawa
Space and Laser Communication Development Division, NEC Corporation
Ikebe 4035, Midori-ku, Yakohama, 226 Japan
and
Shigeru Tanaka
Sumitomo Electric Industries Ltd. Yokohama Laboratory
Taya 1, Sakae-ku, Yakohama, 224 Japan

Abstract

A new phase-lock network using fiber-optic system was developed as a local signal distribution system for 84 antennas of the Nobeyama Radioheliograph. This network is a open loop system and consists of a master oscillator with an E/O converter, a 1-to-84 optical divider, phase stable optical fiber cables and phase stable phase-locked oscillators with O/E converters. Phase stability of the network and phase noise generated at the O/E converter are discussed. This phase-lock network insures the required phase stability of 3deg/6hours. The phase noise increases the coherent loss of 0.1% at the correlator output, which is very low. This is the first large application of fiber optic devices to an open loop phase-lock network. Our system is very simple and phase-stable. Therefore, it is suitable to the connected array with large number of antennas.

1 INTRODUCTION

RF signal transmission systems using optical fibers have advantages of low loss, wide bandwidth and high durability to electro-magnetic interferences. These characteristics are beneficial to long-distance signal transmission. Furthermore, development of a specific optical fiber extended applications of the fiber-optic system to precise timing signal transmission and phase-lock link [1]. At the JPL, a fiber-optic system was tested to distribute a reference frequency to a Deep Space Station, which usually employed coaxial cables [2]. In a VLBI system of the National Astronomical Observatory, Japan, the fiber-optic system is installed to transmit a frequency standard signal from a hydrogen master to a remote antenna. The phase stability was about 55 times better than ordinary coaxial cable transmission systems [3]. At the National Laboratory for High Energy Physics, Japan, a part of coaxial cable link between the 2.5GeV LINAC gun room and the TRISTAN control room was replaced by a fiber-optic system [1]. Another approach using an active phase stabilizer was also developed at the JPL, and it compensated the delay variations using signals reflected at remote units [4]. At the CSIRO, an optical fiber network is installed to the Australia Telescope. This telescope is a radio interferometer with 5+1 paraboloidal antennas of 22-m diameter and a closed loop network with active phase stabilizer is used to lock the phase of local oscillators.

In the radio interferometer, precise measurements of phase among received signals are quite essential to synthesize high-quality images. Although the closed loop phase-lock network is quite stable, its configuration is complicated. As the open loop network has a simple structure, it is suitable for the interferometer with large number of antennas. At the Nobeyama Radio Observatory, a new radioheliograph is now under construction [5]. This system is a radio interferometer for solar observations, which consists of 84 small paraboloidal antennas of 80-cm diameter. These antennas are aligned on a T-shaped baseline of 490m east-west and 220m north-south. In this system, an open loop phase-lock network is installed to synchronize local oscillators. This is the first large application of fiber optic devices to an open loop phase-lock network. The phase stability of the open loop network is sensitive to the phase responses of the devices in the network. In addition, noises generated by the optical devices in the network increase phase noises of local oscillator outputs and degrade the sensitivity of the interferometer. Therefore, we have analyzed the phase stability and the phase noise of this network.

In this paper, we describe the outline of the developed optical fiber phase-lock network and discuss the phase stability and the phase noise of this network.

2 OUTLINE OF THE PHASE-LOCK NETWORK

The Nobeyama Radioheliograph operates at 17GHz. Signals received by the antennas are amplified and frequency-converted to IF signals of 200MHz, and transmitted to a observation building. Then, cross-correlations of the received signals are measured for all antenna pairs to obtain Fourier components of radio images. Figure 1 is a function block diagram of the receiver. Low-noise amplifiers, frequency down-converters and local oscillators are installed in front-end boxes of antenna sites. Each antenna and the observation building are linked through optical fiber cables. The IF signals are transmitted to the observation building via these optical fiber cables. Local signals of frequency down-converters are phase-locked to a reference frequency of 525MHz transmitted from a master oscillator in the observation building through the optical fiber cables. An output of the master oscillator is converted to an optical signal, which is splitted to 84 blanches by optical dividers and transmitted to 84 antennas. The modulated optical signals are demodulated by O/E converters in the front-end boxes. Phase-locked oscillators (PLO's) are used to generate local signals, which are phase-locked to the reference signals.

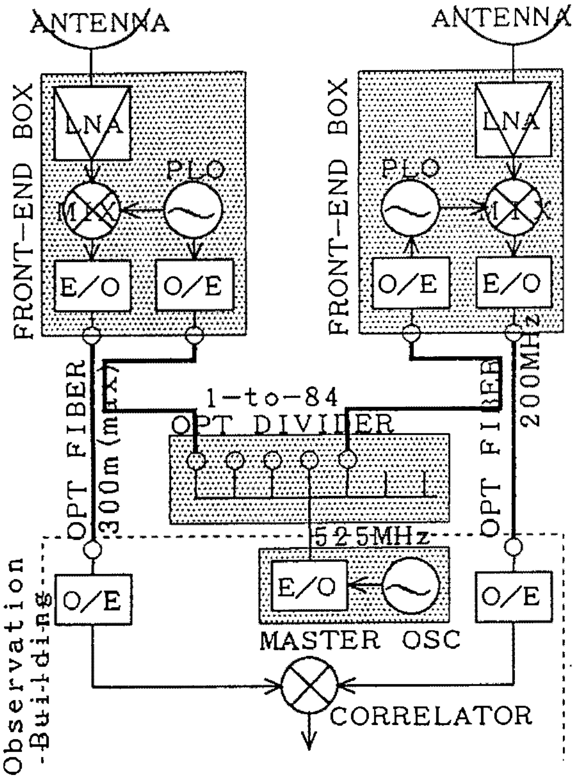


Figure 1: Functional Block Diagram of the Radioheliograph Receiver.

Each PLO of the Radioheliograph consists of a 8.4GHz variable frequency oscillator (VCO), a 1/16 frequency divider, a phase-detector (PD), a low-pass filter and O/E converter. Figure 2 is a block diagram of the PLO. Since the frequency down-conversion to the received signals at 17GHz is performed by a harmonic mixer, the second harmonic frequency of the PLO output is used as the local signal.

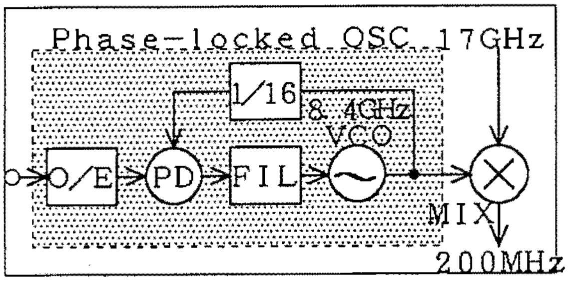


Figure 2: Block Diagram of the PLO.

3 PHASE STABILITY OF THE NETWORK

As mentioned above, the phase-lock network is an open loop system and phase drifts of the network arise from following origins;

(1) temperature responses of the optical fibers,
(2) responses to mechanical stress of the optical fibers,
(3) temperature responses of PLO's,
(4) responses to input power drifts of PLO's, and
(5) responses to supply voltage drifts of PLO's.

Phase drifts caused by (4) and (5) are relatively small.

The temperature coefficient of the phase stable optical fiber cable is less than 1.5ppm/C (5ps/km/C). The cables are buried 1.2m depth in the ground, where the temperature variations are less than 0.1C/day. As a result, phase drifts of 17GHz RF signals are less than 1 deg/day for the cables of 300m length.

Optical fibers are also used to transmit the local reference signal across azimuth and elevation axes from antenna bases to the PLO's in the front-end boxes behind main reflectors. Fibers are bent and twisted according to tracking motions of antennas, with the minimum bending radius of 3 cm. These mechanical stresses cause phase variations of transmitted RF signals. When the cable is bent from the straight line, the phase of RF signals increases according to decrease of the bending radius. Figure 3 shows the relation of the phase changes of RF signals and the bending radius of the cable. Phase change of RF signals at 17GHz is about 10deg when the cable is bent from the straight line to the radius of 3cm. As the cable is loosely bound at the axes to avoid such a strong stress in case of the Radioheliograph, the actual phase variations are much smaller.

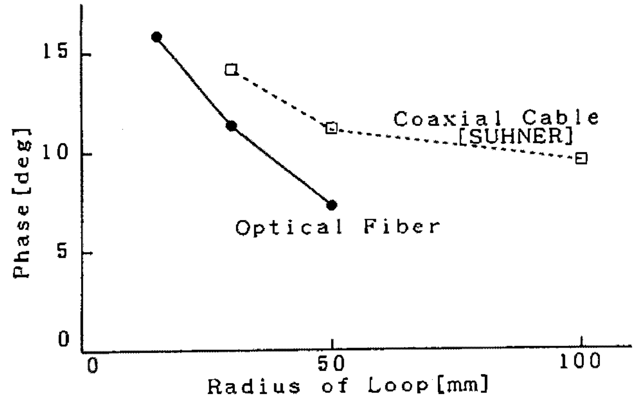


Figure 3: Relation of the Phase of RF Signals and the Bending Radius.

Phase characteristics of the devices in the PLO are sensitive to temperature variations. In the Radioheliograph, these devices are placed on a thermally stable base controlled by a Peltier device. The phase deviations of the temperature stabilized PLO due to ambient temperature variations are 0.5deg/C at 8.4GHz as shown in Figure 4. The PLO is mounted in the front-end box which is maintained at temperature of 40±0.5C.

Phase and gain error in the measured Fourier components cause a reduction in gain of the synthesized beam and an increase in its sidelobe level. Total Phase stability required to the Radioheliograph is less than 3deg during observation time of 6hours. This requirement is sufficiently satisfied in the phase-lock network of the Radioheliograph.

4 PHASE NOISE OF THE NETWORK

The phase noise of the PLO degrades the sensitivity of the radio interferometer. The degradation of sensitivity is evaluated as the coherent loss of correlation output [6]. Usually, the phase noises of the master oscillator and the VCO are main origins of the PLO phase noises. In the new phase-lock network, the additional phase noise is generated in the O/E converter of the PLO. The phase noise level is explained by using the carrier-to-noise ratio (C/N ratio). The output RF signal power of O/E converter is shown by $P_s=\eta eP^2R_L/2h\nu$, where P is the optical input power, η is the quantum efficiency of photo diode, e is the charge of an electron, R_L is the output load register, h is the Plank's constant, ν is the light frequency. In the phase-lock network of the new radioheliograph, the optical input power is about -25dBm including the transmission loss of both the optical divider and the optical fiber cables. The wavelength of light is 1.3μm. The output load register is 2.8kΩ. Therefore, the detected RF signal power is calculated to be 7.9x10^{-9}W, where we assume the quantum efficienc of photo diode η is 60%. The output noise power of the O/E converter is estimated to be about 1.7x10^{-20}W/Hz, which is limited by the thermal noise of the output load register in case of the PIN photo diode. The C/N ratio derived from the above values is -116dB/Hz. On the other hand, the measured value of the C/N ratio was about 101dB/Hz at 10kHz offset point from carrier when the optical input power was -25dBm. The measured ratio is deteriorated by 15dB compared with the calculated one, which is probably caused both by the noise included in the master oscillator output itself and by inadequate matching between the photo diode and the transducer amplifier. Assuming the cut-off frequency of the PLO loop filter is 50KHz, this experimental result corresponds to the additional coherent loss of less than 0.1%.

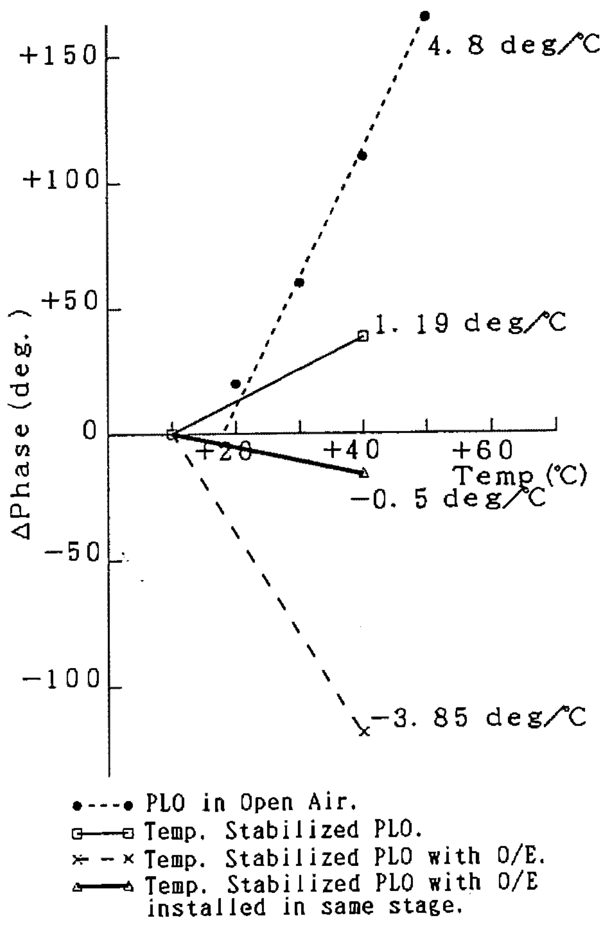


Figure 4: Realtion of the Phase of PLO Outputs and the Ambient Temperature.

5 SUMMARY

An optical fiber phase-lock network is installed to the Nobeyama Radioheliograph. This network is a open loop system and its phase stability strongly depends on responses to temperature and mechanical stress of the optical fiber cables and response to temperature of the PLO's. The network satisfies the phase stability of 3deg/6hours in rms, which is required in the Nobeyama Radioheliograph. The phase noise generated at the O/E converters causes the coherent loss of 0.1% at the correlator output.

6 REFERENCE

[1] S. Tanaka, Y Murakami, Y. Sato and J. Urakawa, "Precise Timing Signal Transmission by a New Optical Fiber Cable," KEK Report 90-9, pp.1-23, 1990.
[2] G. Lutes and L. Primas, "State-of-Art Fiber optics for Short Distance Frequency Reference Distribution," TDA Progress Report 42-97, pp.81-87, 1989.

[3] K. Sato and T. Hara, "Performance of Optical FIbers for Reference Frequency and IF Signal Transmissions in VLBI Observa tions," Proc. of the Twenty-first Annual Precise Time and Time Interval (PTII) Applications and Planning Meeting, pp.421-430, 1989.

[4] L. E. Primas, G. F. Lutes, R. L. Sydnor, "Stabilized Fiber Optic Frequency Distribution System," Internal Report.

[5] H. Nakajima, K. Kai, S. Enome, K. Shibasaki, M. Nishio, T. Takano, C. Torii, Y. Shiomi, H. Sekiguchi, M. Sawa, T. Bushimata, S. Kawashima, N. Shinohara and T. Kosugi, "A New Radio Heliograph at Nobeyama," The 3rd Asia-Pacific Microwave Conference Proceedings, pp.1021-1023, 1990.

[6] A. E. E. Rogers and J. M. Moran, Jr., "Coherence Limits for Very-Long Baseline Interferometory," IEEE Trans. on Instrumentation and Measurement, Vol. IM-30, No.4, pp.283-286, 1981.

Replacing PS Controls Front End Minicomputers by VME Based 32-bit Processors

A. Gagnaire, N. de Metz Noblat, Ch. Serre, Cl.H. Sicard
CERN - 1211 Geneva 23 - CH

Abstract

The PS controls have started the first phase of system rejuvenation, targeted towards the LEP Preinjector Controls.

The main impact of this phase is in the architectural change, as both the front-end minicomputers and the CAMAC embedded microprocessors are replaced by microprocessor based VME crates called Device Stub Controllers (DSC).

This paper discusses the different steps planned for this first phase, i.e:

- implementing the basic set of CERN Accelerator common facilities for DSCs (error handling, system surveillance, remote boot and network access);
- porting the equipment access software layer;
- applying the Real-time tasks to the LynxOS operating system and I/O architecture, conforming to the real-time constraints for control and acquisition;
- defining the number and contents of the different DSC needed, according to geographical and cpu-load constraints;
- providing the general services outside the DSC crates (file servers, data-base services);
- emulating the current Console programs onto the new workstations.

I. INTRODUCTION

The CERN accelerators are composed of two sets: the PS Complex of ten small and fast cycling machines (PS Division), the SL Complex of two big and slow cycling machines (SPS and LEP, SL Division). The rejuvenation of the CERN Proton Synchrotron control system is done on a basis of a common CERN project, aiming at a real convergence between all accelerator control systems. [1] [2]

The first phase of PS control system rejuvenation has started in 1991 for the subset of LPI machines (LEP Pre-Injector). [3] The main impact of the architectural change is the replacement of both front-end minicomputers and distributed CAMAC embedded microprocessors by a set of distributed microcomputers linked on an Ethernet segment with a local file server.

These microcomputers called DSC (Device Stub Controller) are based on both standards PC and VME crate with 32-bit embedded microprocessors. For the PS control system, the VME crates are mainly used. The DSC provides a uniform interface to the equipment and acts as a master and data concentrator for distributed equipment, interfaced via field buses. Due to the large investment in the associated interface equipment, the serial CAMAC loops are kept and their control are done via a serial driver module in the VME crates. [4]

II. BASIC FACILITIES FOR DSC

A. Control System Architecture

The common accelerator control system architecture consists of three layers:

- control room layer with workstations and central servers,
- front-end computing layer distributed around accelerators and based on the DSC,
- equipment control layer with ECA (Equipment Control Assembly) control crates which form part of the equipment.

For the PS Control System the communication between the two first layers, as well as the communication within these layers, is based on a TCP/IP network. The processor of the VME crate has an on-board Ethernet controller as a standard link to the network, and is a diskless machine for reliability [disks proved to be the weak point of the actual LEP PCA (Process Control Assembly)] and because the back-up procedures and management of files and data are supposed to be easier when storage is less distributed. The different programs of the different DSC are centralized on a single server.

B. Choice of a Real-Time UNIX Operating System

The constraints choosing an operating system for the front-end processors were the following:

- the system has to be real-time in order to warranty a predictable response time to external events,
- the same operating system must be available for both VME based MC68030 and PC processors,
- for networking, TCP/IP (with BSD socket interface library) and NFS client are required,
- system must be able to run diskless without swapping,
- shared libraries and data segments are required, as well as source level debugger.

In order to minimize the formation required to develop applications, we had to minimize the heterogeneity between various systems. We had to re-use common facilities developed for the LEP accelerator control (based on Xenix system). This resulted in the choice of the LynxOS real-time

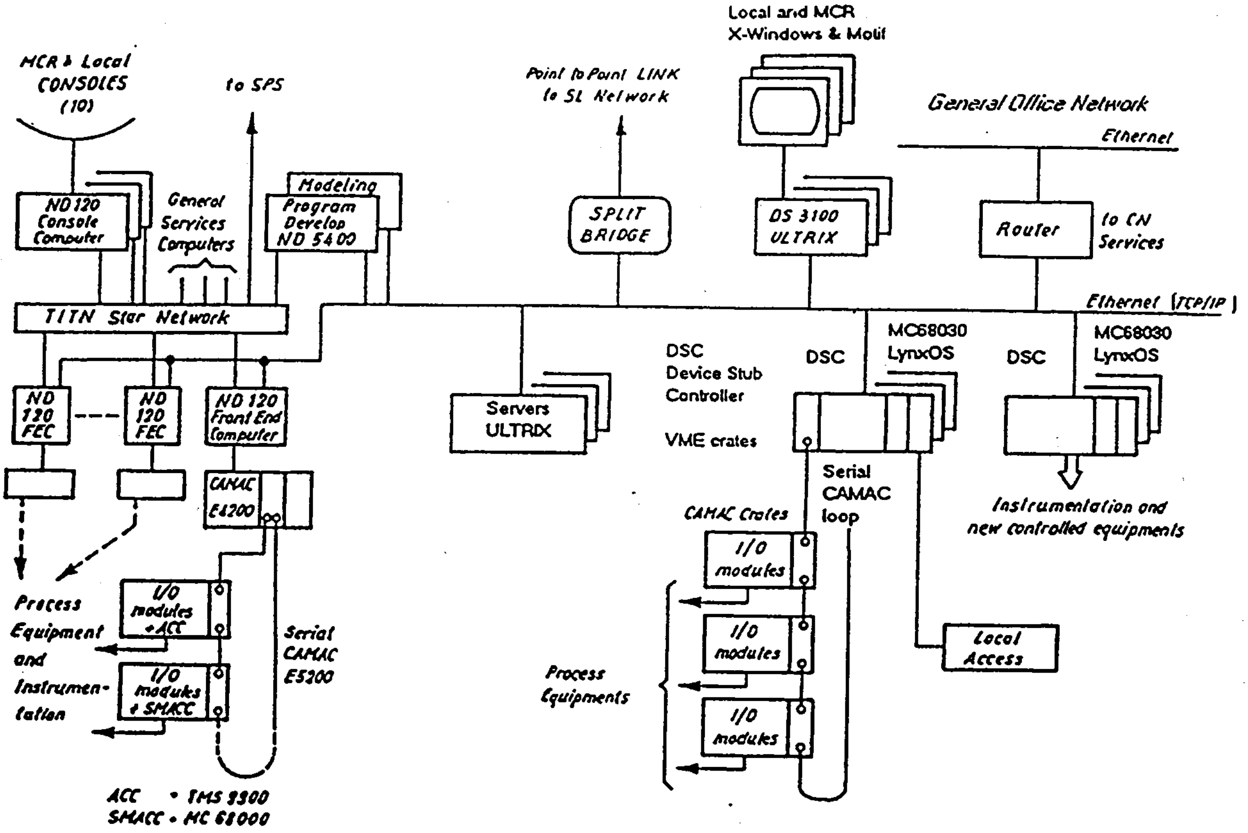


Figure 1. PS Control System during Transition Phase

system, which offers the most recent POSIX standard definitions, especially for the real - time (draft 1003.4 and 1003.4a). The biggest advantage of this choice is that the whole system documentation is intended for a C programmer and even a device driver can be written in C.

C. Common CERN Facilities

One of the main goals of the CERN accelerator control systems convergence is the non-duplication of effort inside the accelerator community. For the front-end processors software the work was shared between both PS and SL Divisions for the following facilities:

- RPC (Remote Procedure Call) and network compiler,
- Common ORACLE on-line Data Base management,
- Error reporting and system survey,
- Nodal interpreter (written in C) running on the different platforms of the network, with network extensions based on the TCP/IP protocol suite, and integrated with X-Windows and Motif environment. [5]

III. PORTING THE PS CONTROL APPLICATIONS TO THE DSC

A. Equipment Access Software

The present application structure layout providing the equipment access from the workstations remains unchanged. This software, actually located in the front end minicomputers and in the CAMAC embedded microprocessors, is ported in the DSC processor. The former code (mainly in C) of the CAMAC embedded processors running RMS68K operating system is, for a major part of it, directly re-used under LynxOS operating system.

In the VME context, simple VME interface boards are accessed using the memory mapped I/O. Connections to field buses, such as serial CAMAC, GPIB or 1553, are handled by standard device drivers. Interface compatibility libraries hide the structure changes from the former implementation. The control data of these equipment access routines use the UNIX "shared memory segment" and the code of all the application programs call a shared library where all the equipment access routines are stored. This point will increase the reliability as the programs use a single copy of the library.

B. Real-Time Tasks

Whenever possible, the same interface library used for equipment access is re-used by real time tasks. The application programs are not integrated at the level of interrupt routines, but are working at the level of the user task. In case of response time problems, the system thread facility will be used. Our main real time constraints are today repetitive acquisition at the rate of 3 msec. The actual measured response time for such a processing (including the wake up of a task waiting for an external event, the time for this task to access the CAMAC interfaced equipment through a standard device driver) is compatible with these constraints.

C. General control applications

Detection of equipment alarms has been incorporated into the equipment access software layer, through a unique function, which can return an alarm indication, for all classes of equipment.

Application libraries and tasks may log informative or fault messages to a central service, which will record them in the ORACLE data base and, if they occur in a remote call, display them on a window of the originating workstation.

A general data collection mechanism, synchronized with the accelerator cycle, allows the workstations to subscribe to specific sets of control parameters, and thus receive regularly complete updated data messages.

IV. FIRST ACTUAL IMPLEMENTATION IN THE PS COMPLEX

First actual implementation of the rejuvenated common control system in the PS Accelerator Complex concerns the LPI machines (Linac e^-, Linac e^+ and electron/positron accumulator). A present console will be replaced by a work place of three DEC workstations running ULTRIX, associated with selection and observation of analog and video signals. The present control of the LPI equipment is done through front end minicomputers (NORD 120) and CAMAC embedded microprocessors: they will be replaced by a file server and a set of DSC, each of them driving a serial CAMAC loop, linked on a regional Ethernet segment of the control network. (Fig.1)

For the man machine interaction, at the level of the workstations, in order to re-use as much as possible the interactive application software, the Nodal functions of the present consoles are emulated on the workstations in order to minimize changes to existing Nodal programs. The generic programs developed in C for the workstation interaction prototype will be integrated (knob control, synoptic presentation, console manager). [6] [7] No accelerator real-time constraints must be treated at the level of the workstations, the DSC will take care of them. The number of DSC is function of:

- the geographical distribution of the CAMAC crates and the topology of the equipment interfaces,
- the real-time load of each DSC processor, depending on the real-time constraints of the application programs controlling the equipment (instrumentation, pulse-to-pulse modulation).

As often as possible, one single DSC is used to control one given beam instrumentation device; this point facilitates the device diagnostic, and allows more development capability to the equipment specialist.

As one of central service given by the network server, one powerfull server is used to house the ORACLE on-line data base management service; this data base is also used for the off-line data preparation to allow an adequate continuous running of the application whatever happens to the data base server.

V. CONCLUSION

The main points introduced by the common rejuvenated control system for the CERN accelerators could be listed as follows:

- UNIX is the single operating system for all the control levels of the architecture,
- the network is based on commercial well established standards as TCP/IP protocol suite and NFS facilities,
- to increase the reliability and to ease the maintenance, the workstations and the DSC are used as diskless devices,
- the access to equipment is done through a single homogeneous level, the DSC,
- the present important hardware investment (serial CAMAC crates for the PS Complex) is kept; the equipment interface does not need to be modified immediately,
- we hope that the proposed architecture, based on open and well accepted standards (hardware as well as software) will permit continuous upgrading as the technology evolves.

VI. REFERENCES

[1] The PS and SL Control Groups, PS/SL Controls Consolidation Project, Technical Report, CERN PS/91-09 (CO), CERN SL/91-12 (CO), April 1991.

[2] R. Rausch, Ch. Serre (Editors), Common Control System for the CERN PS and SPS accelerators, Nov. 1991, these Proceedings (ICALEPCS, Tsukuba, Japan, 1991).

[3] G. Benincasa, G. Daems, B. Frammery, P. Heymans, F. Perriollat, Ch. Serre, Cl.H. Sicard, Rejuvenation of the CPS Control System: the first slice, May 1991, Proceedings of 1991 Particle Accelerator Conference, San Francisco.

[4] W. Heinze, Driving serial CAMAC systems from VME crates, Nov. 1991, these Proceedings, (ICALPECS, Tsukuba, Japan, 1991).

[5] F. Perriollat, G. Cuisinier, A. Gagnaire, Nodal in C, PS/CO/note 91-20, Oct. 91 (CERN internal note; not published).

[6] M. Boutheon, F. Di Maio, A. Pace, General Man-Machine Interface used in accelerator Controls: some applications in CERN/PS Control system rejuvenation, Nov. 1991, these Proceedings, (ICALEPCS, Tsukuba, Japan, 1991).

[7] P. Antonsanti, M. Arruat, J.M. Bouché, L. Cons, Y. Deloose, F. Di Maio, Workstations used as Consoles for the CERN PS Complex: Setting up of the environment, Nov. 1991, these Proceedings, (ICALEPCS, Tsukuba, Japan, 1991).

Device Controllers using an Industrial Personal Computer of the PF 2.5-GeV Electron Linac at KEK

Yuji Otake, Mitsuhiro Yokota, Kazuhisa Kakihara,
Yujiro Ogawa, Satoshi Ohsawa, Tetsuo Shidara and Kazuo Nakahara

National Laboratory for High Energy Physics
1-1, Oho, Tukuba-shi, Ibaraki-ken, 305, Japan

Abstract

Device controllers for electron guns and slits using an industrial personal computer have been designed and installed in the Photon Factory 2.5-GeV Electron Linac at KEK. The design concept of the controllers is to realize a reliable system and good productivity of hardware and software by using an industrial personal computer and a programmable sequence controller. The device controllers have been working reliably for several years.

I. Introduction

Operators of the Photon Factory 2.5-GeV Electron Linac (PF Linac) were required to reduce the beam tuning time for starting up; therefore, detailed information concerning the accelerator was necessary in order to understand the behavior of the linac. New device control systems, including slit controllers of the beam energy analyzing system and electron gun controllers for the PF Linac, have been installed for improving the operational performance, such as monitoring the electron gun and beam parameters, since 1989.[1]

If we consider the configuration of a device controller for an accelerator, combining a personal computer and a programmable sequence controller (sequencer) is the best solution. This is because they have the advantage of low-cost and good productivity for a device controller. Furthermore, they are now very popular and reliable, and have many cheap circuit boards and extension units as a digital/analog I/O. A personal computer complements some of the functions of a sequencer, such as the display of data and the management of data/program file. For this reason, industrial personal computers (NEC FC-9801V) and sequencers (OMRON C200H) were employed for the device controllers. The FC-9801V has been improved in reliability, compared with the usual personal computer, like the PC-9801, and can run on BASIC encouraging non-expert programers. On the other hand, the sequencer has also been improved regarding its immunity to bad environmental conditions.

For connecting the device controllers to the PF Linac control system, a communication board with a CPU was developed so as to be used in the industrial personal computer. The board separates communication tasks from the main CPU (CPU of the industrial personal computer), and effectively increases the system reliability.

In this paper, we describe the electron gun and slit controller systems according to the above-mentioned idea, and give a brief description of the PF Linac control system, since these device controllers act as a front-end of this control system.

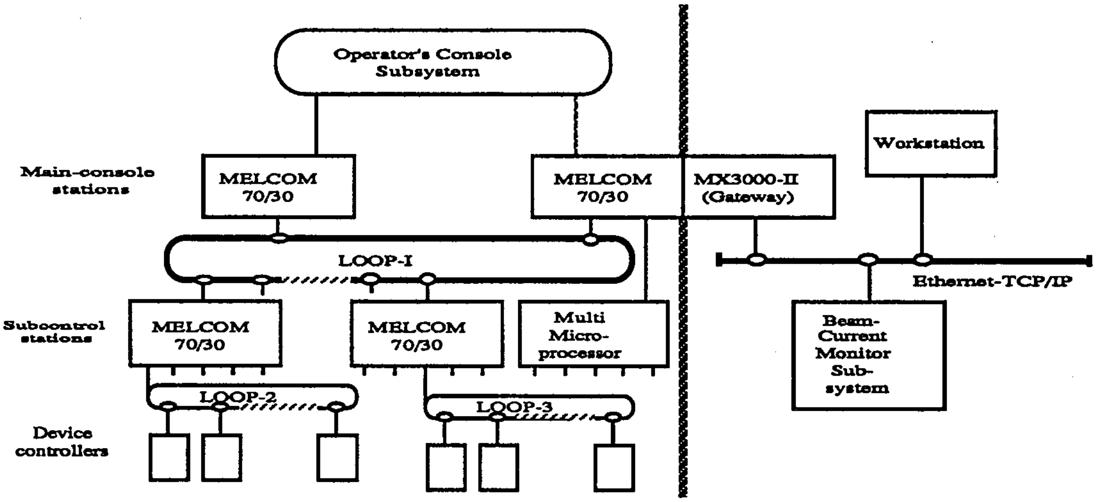


Fig. 1 Block Diagram of the PF Linac Control System

II. Control System of the PF Linac[2]

The control system of the PF Linac shown in Fig. 1 has three kinds of computer communication loops: Loop-1, Loop-2 and Loop-3. Loop-1 is an optical communication link used for the main minicomputers (Mitsubishi, MELCOM 70/30); Loop-2 and Loop-3 are optical communication links between satellite MELCOM 70/30 and the microprocessor-based device controllers.

The device controllers reported here are connected to the minicomputer system through Loop-3: asynchronous communication at a signalling rate of 50 kbit/s is possible.

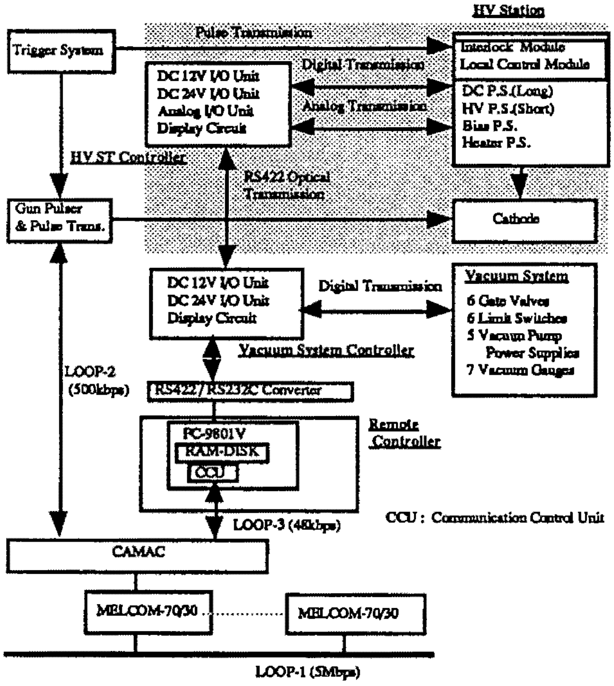


Fig. 2 Block Diagram of the Electron Gun Control System

III. Electron Gun Control System[3,4]

The device controller for the electron gun is basically constructed using an industrial personal computer (FC-9801V) and two sequencers (C200H), as shown in Fig. 2. The sequencers are used to control a high-voltage dome and a vacuum system for the electron gun. The controlled parameters of the sequencer for the high-voltage dome are the grid voltage of a long-pulse mode (1 µs electron beam) and a short-pulse mode (4 ns electron beam); a grid bias voltage; a heater voltage and the heater current of the cathode as an analog voltage control device; and ON/OFF for the devices and the interlock items as a digital control device. The vacuum system controller mainly controls digital parameters, such as the open/close action of the gate valves, the ON/OFF for vacuum pumps and interlock items. It also includes a few analog parameters, such as reading the vacuum gage. Usually, the digital parameters are indirectly controlled through a relay circuit from the sequencer.

The optical communication link (9.6 kbps) is based on the RS422 specification and connects the FC-9801V with the sequencers. The individual controllers must be coupled tightly, because one controller's information is important for the others. For example, if the vacuum of the electron gun is destroyed, the sequencer for the high-voltage dome must quickly turn-off the heater power supply of the cathode; otherwise, the cathode surface could be easily damaged by the poor vacuum. The optical link is used to isolate the circuits in the high-voltage dome (about 100 kV) from the ground level.

The FC-9801V, including the loop-3 communication board, has the following roles: One is a gateway for connecting the electron gun control system to the PF linac control system. It interchanges information between the sequencer system and the minicomputer system while converting each protocol. Another is to control the circuits in the high-voltage dome from the ground level. The control software for these functions were written in BASIC.

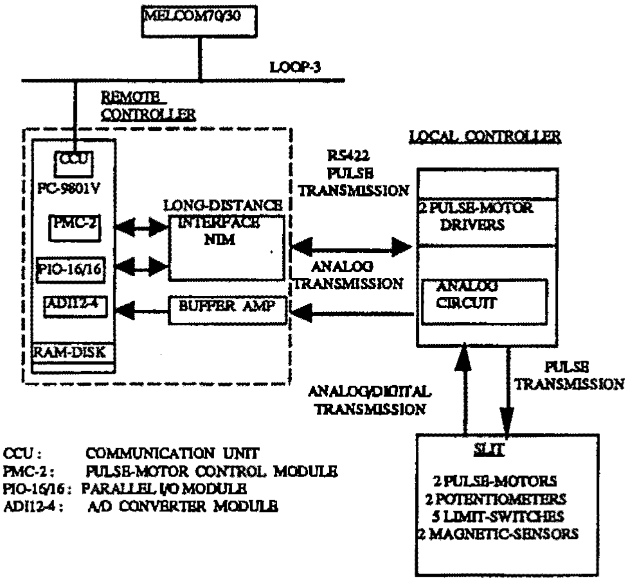


Fig. 3 Block Diagram of the Slit Control System

IV. Beam Slit Control System[5]

The beam-slit control system of the beam-energy analyzing station comprises a remote controller at the klystron gallery and local controllers at the linac tunnel. A block diagram is shown in Fig. 3. The slit controller controls two pulse motors, two potentiometers for position detection and five limit switches which detect the limitation and the collision of moving slit blocks. The FC-9801V is used as a remote controller. One remote controller directly controls a couple of local controllers for slits utilizing an extended pulse motor control board of the FC-9801V. The pulse motor control signal generated in the remote controller is directly transfered to the local controller through the RS422 level signal interface.

The local controller includes the pulse motor drivers, which are directly connected to the slits. It also has a panel for making the silts locally controllable: control buttons for the right and left movement of slits and a position display using a digital volt meter, which shows the potentiometer voltage. The control sequence in the local mode is achieved by a logic circuit (TTL circuit) without any control software.

The control software on the remote controller was written in BASIC. The remote controller also has a control panel with a CRT display and hardware buttons controlled by the software.

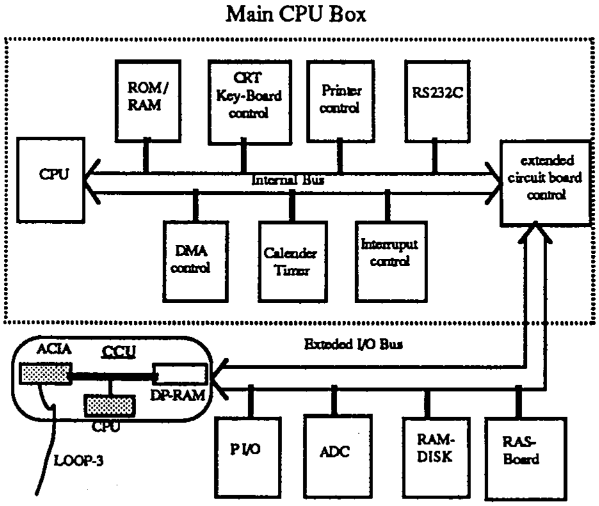


Fig. 4 Configuration of FC-9801V

V. Components of the Device Controllers

A. Configuration of the FC-9801V[6]

The FC-9801V comprises a CPU box and an expansion unit with extended circuit-boards, such as a RAM disk (1 MBytes) with a battery back-up, a process input /output-board (PI/O), an analog/digital converter-board (ADC), a Loop-3 communication-board and a RAS-board. Fig. 4 shows the system block diagram of the FC-9801V. The CPU box within the dotted line in Fig. 4 contains a CPU, a ROM/RAM-unit, a DMA control-unit, a Calender/Timer-unit, a CRT/Keyboard control-unit, a printer control-unit, an Interrupt control-unit and an RS232C-unit.

The FC-9801V is an improved version of a conventional model of the PC-9801; it has better reliability for industrial use. It has good environmental capabilities, such as a wide temperature allowance; an improved physical shock allowance; and improved immunity against noise from the power line. It also improves the self diagnostic functions: a watchdog timer, an over voltage check for the power supply, a temperature emergency alarm, a memory parity check, etc. The RAS-board has the above-described functions. Details concerning the improved functions are shown in Table 1.

The operating system of the FC-9801V is MS-DOS, and the software for it was written in MS-DOS-based N88-BASIC. The MS-DOS-based software system encouraged the rapid development of the control program. Furthermore, the auto-starting-up function for the FC-9801V was established by the RAM disk and by defining the AUTO-EXECUTION BATCH file of the MS-DOS system. Starting-up without a floppy disk or a hard disk has great advantages regarding reliability.

* RAS Board Function	
1, Detection of the Power Supply OFF.	-> NMI
2, Memory Parity Check.	-> NMI
3, Watchdog Timer.	-> NMI
4, Detection of the Temperature alarm.	-> NMI or INTR
5, External Alarm.	-> NMI
6, Remote Reset.	

(NMI, Non-Maskable Interrupt / INTR, Maskable Interrupt)

Table 1 RAS Board Functions

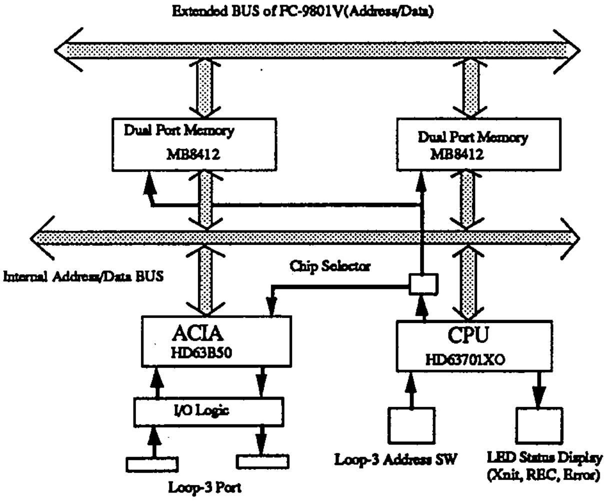


Fig. 5 Configuration of the CCU

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S11LLC02

B. Communication Control-Unit(Loop-3 CCU)

The communication control-unit(CCU) interfaces between the device controller and the PF linac control system. Fig. 5 shows the configuration of the CCU board, which has a one-chip CPU(HD63701) with ROM and I/O ports, an asynchronous communication interface(ACIA) and dual-port RAMs. The dual-port RAMs are used to exchange control data between Loop-3 and the FC-9801V.

The BASIC program on the FC-9801V can easily handle the CCU through the mail-box on the dual-port memories. Separating of the communication task from the CPU on the FC9801V to the CCU board is extremely effective for improving the reliability and productivity of the software.

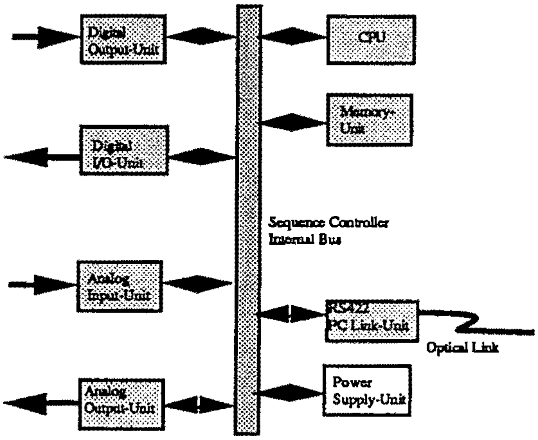


Fig. 6 Sequencer System

C. Sequence controllers[7]

For designing the device controllers, great care should be paid to the large noise caused by the high-voltage pulse (about 100 kV and 2 µs) of the gun pulser. Therefore, the sequencer was used for the device controller; it has a CPU, an analog I/O, a digital I/O and a communication link and power supply units. A block diagram of the high-voltage dome controller is shown in Fig. 6. The capability of the individual units for the environmental acceptance has been improved as in the case of the FC-9801V. For example, the digital I/O-unit uses an optical coupled I/O, and the analog I/O-unit uses a differential I/O.

The program for the sequencer can be written as a ladder circuit diagram, which has been widely used to design a relay circuit. Furthermore, it can be handled and programmed on the FC-9801V. We can examine the sequencer's program by on-line checking from the FC-9801V. This is very convenient for reading, writing and checking programs.

VI. Conclusion

The system now works well and is reliable. By utilizing a commercially available system of an industrial personal computer and a programmable sequence controller, the electron gun and slit controllers were made. The installed battery back-up RAM disk, especially, made the maintenance easy and removed a mechanical unreliability which occurs during a program boot from a floppy disk or a hard disk; furthermore, it improves the starting-up time of the FC-9801V. Since we have adopted commercial devices and BASIC language, the device controllers could be developed.

In the future we will use a real-time multi task operating system for the device controller, and will replace local controllers of slits with sequencers.

Acknowledgement

The authors would like to thank Prof. Gen'ichi Horikoshi and Prof. Akira Asami for their support, as well as the members of the PF Linac.

Reference

[1], T. Shidara et al., "Improvement on Monitor System in the KEK 2.5-GeV Linac", KEK-Report-88-11, January 1989.
[2], K. Furukawa et al., "Recent Progress in the Control System of the KEK 2.5-GeV e^-/e^+ Linac", Nucl. Inst. and Meth. A293(1990)16-22.
[3], M. Yokota et al., "New Power Supplies for PF Linac Electron Guns and their Control", Proc. of the 16th Linac Meeting, Nihon Univ., P. 212-214, September 1991.
[4], "KEK Photon Factory Activity Report", 1990.
[5], K. Kakihara et al., "Energy Slit System for the PF 2.5-GeV Linac", Proc. of the 7th Symposium on Accelerator Science and Technology, P. 231-233, Osaka, December 1989.
[6], "Hardware Manual of FC-9801V in Japanese", (NEC, 1989).
[7], "Hardware Manual of C200H in Japanese", (OMRON, 1989).

High accuracy ADC and DAC systems for accelerator control applications.

E.Kuper, A.Ledenev
Institute for Nuclear Physics,
Novosibirsk, 630090, USSR

Abstract

In the work presented here the ways of construction, the apparatus for the precision measurements and control systems incorporated in the accelerating facilities of INP are considered. All the apparatus are developed and manufactured in the standard of CAMAC.

Introduction

While carrying the experiments on the precision measurements of the mezon masses on the installations with the electron - positron colliding beams one has to use the apparatus of a class 0.001% with the resolution about 0.0001%. An instability of the main power supply sources of magnetic systems of storage rings should not exceed 0.002%.

The powerful RF generators, the controlled sources of power supply with an output power of a few hundred kilowatts, pulse components of electron-optical channels, numerous digital devices including computers are the sources of different kinds of noises. Under these conditions, the stronger requirements on the noise damping are posed to the measuring and control equipment and to the analog data transfer lines.

In the power supply system of the facility VEPP-4 one has to measure of about two thousand points and to form the control signals for more than 500 channels. The time of energy rise is of a few tens of seconds. In the mode of operation one needs the high accuracy matching (0.1% - 0.01%) in the field variation in the magnetic components of accelerators. The technical parameters of the control and measuring structures should provide the operation of the power supply systems both in the static and in dynamic modes of operation.

Digital - to - analog converters

Usually, the power supply sources of the storage ring facilities requires the digital-to-analog converters (DAC) of quite a low fast action at an accuracy ranging from 0.1% up to 0.001%. Therefore, most of the converters used are designed on the base of the pulse width modulation PWM. The advantages of the given type of DAC are well known: the minimum of precise components at a practically arbitrary resolution, high linearity, easily achieved the galvanic isolation of the analog part and consequently, a low price.

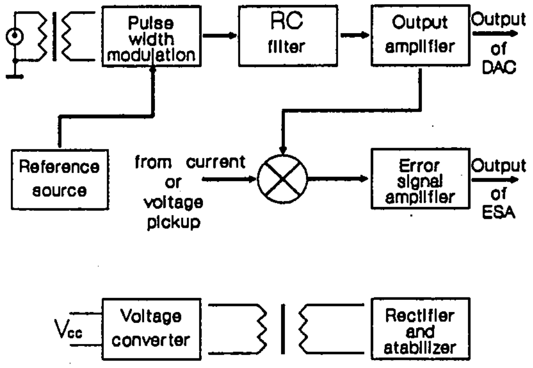


Fig.1. The analog section of PWM-DAC.

One of the popular developments of INP is an 8-channel code-to-duty factor converter (CDFC) located in the crate and transferring the control signal to DAC - PWM integrated directly to the control objects. The DAC signals in the form of different polarity pulse, the distance between carrying the data on reference voltage for control system are transferred through the coaxial cable with the transformer de-coupling to the distance of up to 500 m. The simplified schematic diagram of the converter is given in Fig.1. The pulsed signals from DAC arrive at the trigger controlled by the analog switches. The PWM modulated signal is filtered with the RC filter of the 3rd order. In order to match it with the control system the error signal amplifier (ESA) is envisaged which equalizer the DAC output signal to that from the current or voltage sensor. The galvanic de-coupling on power supply is performed with the help of high frequency converter with the transformer of special design with the minimum crossing capacitance. DAC parameters are: 16 bites, error - 0.01%, settling time - 0.4 s, temperature factor of the output voltage - 0.0003%/K. The given configuration is being widely used in the systems of pulse power supply of the transport channels for charged particles, in the power supply sources for the "high current" correctors, i.e. in those cases, where the controlled objects are distanced considerably and their groundings are explicitly non-equipotential.

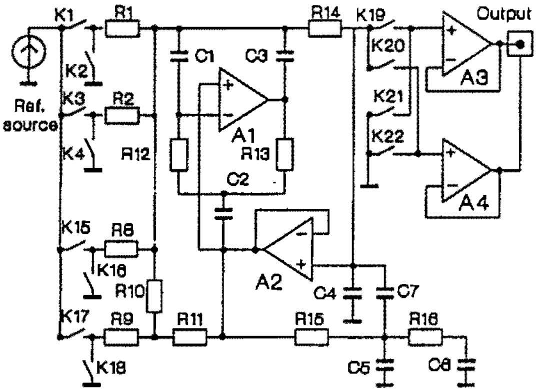


Fig.2. The analog section of multiphase PWM-DAC.

For the problems with higher requirements to the control accuracy (power supply systems of the main magnets and lenses of storage rings) the precision DAC was developed that is based on the use of the multiphase pulse-width modulation. This method is the improved version of the PWM. Its use enables one to reduce the settling time by a factor of the number of phases with respect to the conventional case and similarly to reduce the switching frequency. In this case, the requirements to the fast switching time of the analog switches determining the PWM signal became weaker. This fact enables one to simplify substantially the technical-design solutions of the switches and drivers using the standard logic elements of CMOS kind. The schematic diagram of the analog part of the apparatus is given in Fig.2. DAC is performed according to the two-cascade circuit. The output voltage is the sum (with weights 1/2048) of voltages of two independent DACs. The first one converts 11 senior bits and has an 8-phase generation of the output signal (switches K1 - K16). The second DAC for junior bits is a single-phase, 8-bit (K17 - K18). The voltages of both the DACs are summed by the resistors and smoothed by one filter of low frequency (A1, A2). The bipolar voltage is produced by the output circuit (K19 -K22, A3, A4). In the source of output voltage the precision reference diod is employed in the oven with the stability of temperature 1K.

The apparatus has the following parameters:

scale length	20 bits
voltage range	8,192 V
quantization step	15,625 μV
error (for 3 months)	0.001%
nonlinearity	0.0001%
temperature factor of output voltage	0.00002%/K +2μV/K
settling time with error 0.001%	0.1 s
analog part capacity with respect to the body	150 pF
module width	1 M

In the process of energy retuning of the storage ring of charge particles the matched variation of parameters is required for many power supply sources of magnetic elements with its high accuracy and highly synchronous. In the power supply systems of first generation this problem was solved by the appropriate selection (taking into account the individual characteristics of magnets) of special RC-filters on the DAC output. At present, the most relevant solution of the problem given is the use of DAC with the built-in digital interpolators. Two types of converters have been developed.

The multichannel DAC provides the conversion of a 16 bits code with the error 0.01% over 16 channels. The built-in processor makes the simultaneous variation of the output voltages. The variation law for each channel is given with the intermediate values by which the portion-linear interpolation is performed. Up to 80 intermediate values can be given for each channel. In addition, in each linear part the interpolation time is given within the range from 1 to 63 s with the quantization step of 1 s or in the range from 0.1 to 6.3 s with the quantization step of 0.1 s.

For the control of the precision channel a 20 bit DAC was developed with the built-in signal error amplifier. The conversion error is 0.001%. The functions of the digital interpolator are similar to that described above.

The converter control is performed through the controller with the protocol MIL-STD 1553B. The controller is designed in the CAMAC standard.

Apparatus for measurements of direct voltages

As already mentioned, the real operation of any large physical facility is usually accompanied by the generation of a large wide range of very different electromagnetic noises and quite often the amplitude of these noises is substantially higher than the signal to be measured. Therefore, in the analog-to-digital converters the most noise-resistant method of preliminary integration of the output signal is used.

Using various modifications of the method: the dual-slop integration, multi-tact integration, the method of dynamic integration, the series of ADCs is designed at INP aimed at the use in the multichannel noise-resistance measuring systems. Let us consider briefly the features of construction and parameters of the most characteristic versions ofthis series.

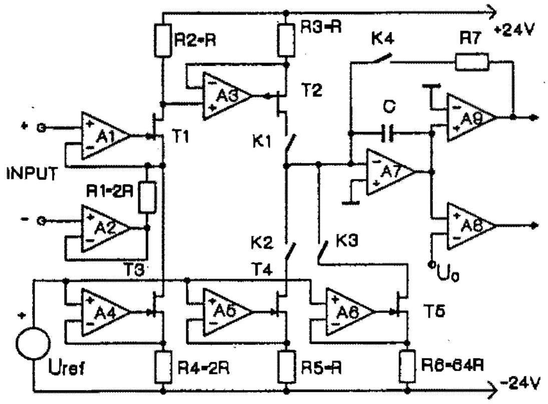


Fig.3. Three-step integrating ADC.

The three-step integration with respect to the dual-slop one enables the reduction of the error that is due to the noises of the integrator amplifier and comparator. An additional advantage of the method is its high resolution at comparatively low frequency of the tact generator. The simplified block diagram of the device is given in Fig.3. The input signal with the help of amplifiers A1 and A2 is converted into a current. A3, T2 is the current mirror. To achieve the fast action the switches K1 - K3 are made with diodes. The main reference signal is generated by the current generator on A5, T4. The reference signal of the third step 64 times lowered is formed with A6, T5. During the first step the integration of the input signal is performed. During the subsequent two steps the integration of the main and divided reference signals is performed. This method is realized on the ADC 15 - 256. Its main parameters are given in Tab.1. The device has a built-in memory for 256 words and the control circuit for the analog multiplexer that transfers the address of the measured channel in the subsequent code through the socket on the front panel. While performing the multichannel measurements the operation is performed in the following way: preliminary to the service register of ADC the initial and final addresses of channels are written along which the measurements should be performed. By the start command ADC performes the given series of measurements and writes the results into the corresponding cell of the built-in storage. The presence of this given mode enables one to reduce substantially the load of the CAMAC data bus in the measurement system.

The method of multistep integration enables one to reach the high resolution and linearity of the converter at quite high fast action. The block diagram of the method performance is given in Fig.4. Its essence is the following. Simultaneously with the input signal integration the reference signal is integrated by a certain algorithm. When the integrator output voltage reaches its threshold value (either A2 or A4 comparator is operated) to the integrator input the reference voltage is applied whose polarity is opposite to the input voltage for the fixed period of time, i.e. the multi-step integration of the reference and input signals is performed. After input signal integration the operation algorithm is the same as that in the previous method. The end of the 3rd step, during which the reduced reference voltage is integrated, is defined by the comparator A3. In this scheme the integrator transfer factor is approximately the same as the number of integration steps when measuring the input maximum signal.

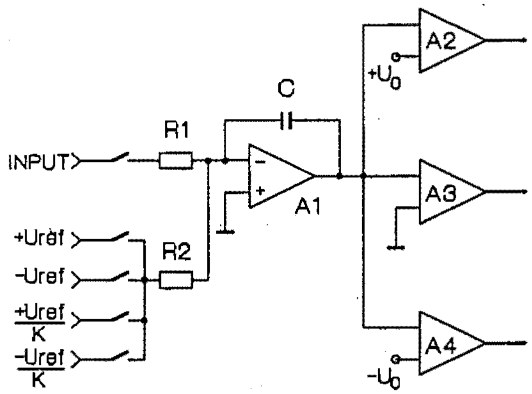


Fig.4. ADC with multicycle integration.

Note that conversion errors related to the polarization of integrator capacitor dielectric are reduced similarly. On the base of this method the precision converter ADC-22 was designed (Table 1). Let us give some additional characteristics of the device that are important for the construction of measuring systems of high accuracy: integration time of the input signal - 20 ms; settling time for the input amplifier (with an error 0.001% - 8 ms; signal measurement range 0.1; 1; 10; 100; 1000 V; resolution capability, respectively, 0.1; 1; 10; 100; 1000 μV; relative error of conversion in the range of 10 V for 8 hours - 0.0005% of the scale, an additional temperature error - 0.00005%/K; temperature drift of voltage - 0.03 μV/K; input current - lower than 10 pA; the input resistance on lower ranges - higher than 100 GOhm.

The method of dynamic integrator. This method is the version of the pulse width modulation PWM conversion with the pulse feed back. The block diagram of the dynamic integrator operation is given in Fig.5. The input signal is applied to the input of integrator A1 through resistor R2. Depending on the integrator input voltage polarity, defined by the comparator A2, an appropriate reference voltage is applied to the integrator input through the switches K1 and K2. (The operation of these switches is synchronized with the switching frequency by the D-trigger.) The specific feature of the method is that simultaneously with the input and reference signals through the capacitor C2 to the integrator input the periodic voltage is applied of the rectangular shape whose amplitude exceeds the sum of modules of the input and reference voltages. The main advantage of the method is that it enables one to vary the time and bits of conversion. This property is especially useful for ADC designed for the multichannel measurements: measurements with not very high accuracy are performed quickly but those of high accuracy - slower but with larger number of bits. With larger time of measurement an additional filtering is envisaged for high frequency noises in the measured signals.

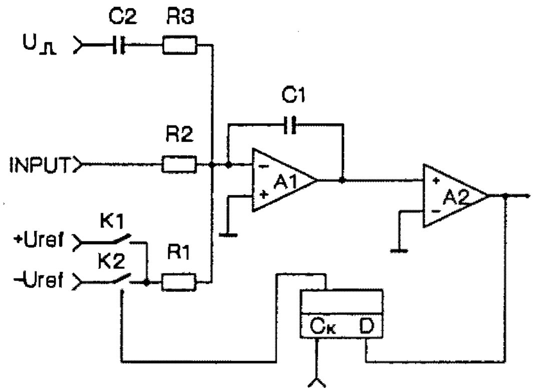


Fig.5. ADC with dynamic-integrator.

On the base of the method of dynamic integrator ADC-20 and ADC-20-256 (Table 1) are designed. Each of these modules has 8 time ranges of measurements (when switching the ranges the scale length changes respectively) and 2 ranges for the input signals (8 V and 500 mV) which enables the measurement of voltage in the microvolt range. ADC-20-256 has a built-in memory and the control for the multiplexer.

Table 1

Type of device	ADC22	ADC20	ADC20-256	ADC15-256
Conversion time, ms	40	7.5-480	1.25-160	0.1
Scale (binary)	22	14-20	13-20	15
Error (for 3 months)	0.001%	0.01%	0.01%	0.01%
Memory (words)	-	-	256	256
Common mode rejection, db	140	120	120	80

The metrological characteristics of devices remain the same within the range 20 K - 50 K.

For the arrangements of the multichannel measuring systems some analog multiplexers been developed at the Institute:

- AM-16R and AM-128R: with 16 and 64 channels respectively, the commutation elements - sealed contact reed relay with switching time of 1 ms, maximum voltage - 200 V, commutation error - 50 μV.
- AM-16RM and AM-32R: with 16 and 32 channels respectively, the commutation elements - thermocompensated reed-relays, commutation error - 1 μV. They are designed for the measuring systems in the microvolt range.
- AM-128: 64 channels, produced based on microcircuits with the complementary MOS-transistors, switching time - 10 μs, input voltage range - 10 V, commutation error - 100 μV.

The multiplexing of input signals is usually performed according to the two wire circuit.

In all the versions the protection is envisaged against the overvoltages by the input. The address register of multiplexers have 8 bits, that enables the union of up to 256 measuring channels per one converter.

Driving Serial CAMAC Systems from VME Crates

W. Heinze
CERN 1211 Geneva 23

Abstract

Large control systems in the 80's were often based on Serial CAMAC loops driven by 16 bit minicomputers. These 16 bit computers, becoming obsolete in the 90's, are advantageously replaced by VME crates. To maintain the investment in Serial CAMAC hardware and software, an inexpensive Serial Highway Driver has been developed which operates in a VME crate as simple I/O module. With this system, both classical configurations, i.e. the Highway Driver on the I/O bus of the minicomputer and the Highway Driver in a so-called CAMAC mother crate, can be replaced with minimal costs and improved performance.

This paper presents a VME Serial CAMAC Driver and compares the performance of the VME driven Serial Highway to the ones driven by minicomputers. The comparison is based on the experience gained with the beginning of the replacement of Norsk Data minicomputers by VME crates in the CERN/PS control system as described in [1].

I. INTRODUCTION

The control system of the CERN Proton Synchrotron complex (PS) is at present based on Norsk Data 16 bit minicomputers which control 26 Serial CAMAC loops connecting approximately 240 CAMAC crates to drive the accelerator equipment. The rejuvenation of the PS control system starts with replacing the minicomputers by VME crates, called Device Stub Controllers (DSCs). The VME computing element is the MVME147 module, performing not only the tasks of the minicomputers but also executing the real-time tasks of the currently used Auxiliary Crate Controllers (ACCs).

The interface between the accelerator equipment and CAMAC has to be maintained for another some 10 years just because the investment in terms of money is so high. This was the reason to develop an inexpensive Serial CAMAC Highway driver as a VME module (abbreviated as SDVME). The first series of 20 SDVMEs have been assembled at CERN but the module is now also fabricated by a major CAMAC/VME manufacturer (C.A.E.N., Viarregio, Italy).

II. DESIGN OF VME DRIVEN LOOPS

The actual Serial CAMAC loops will be rearranged into smaller loops. Every loop is then controlled by one DSC. The new loops consist of up to about 10 CAMAC crates, the number depends on the necessary computing power to control the associated equipment because the CAMAC crates do not house computing elements (the ACCs) anymore. They are not more than simple input/ output devices.

In the present control system, 2 types of CAMAC Serial drivers are used which are now replaced by the SDVME: one on the I/O bus of the minicomputer and one in a CAMAC crate which in turn is driven by a dedicated CAMAC Branch driver. Especially for the latter case which is used in 22 (out of the 26) loops, a considerable overhead of CAMAC equipment and cabling is avoided thus improving reliability, maintainability and equipment access times.

In general, the transmission speed on the Serial Highway of 2.5Mbit/s, bit serial, is maintained. Bit serial transfer permits to continue to use U-port adapters reducing the necessary number of twisted pairs in the Highway cable from 4 to 2 (one for the command, the other for the reply part) and allowing loop reconfiguration. Figure 1 shows the standard configuration. For special cases (instrumentation), the transmission speed is selected to 5Mbyte/s, i.e. byte serial. This is not a problem because in these cases, normally only one CAMAC crate is controlled by a DSC, and the distance is very short.

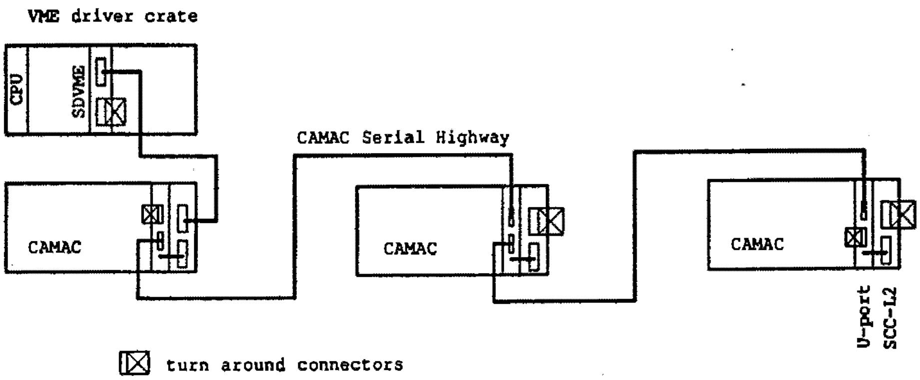


Figure 1. Standard loop configuration

III. CHARACTERISTICS OF THE SDVME

Communication between the SDVME as VME slave and the processor as VME master is governed by the Data Transfer protocol described in the VME specification. It is based on programmed I/O operation, short addressing and 16 bit data width (A16/D16). Driving the Serial Highway is done in conservative mode which simplifies error treatment and recovery.

The transmission speed is switchable to 0.5, 1, 2.5 and 5 MHz for byte and bit serial operation. Conform to the EUR 6100e specification, the appropriate number of space bytes is inserted into the Command messages to guarantee the completion of the Dataway cycle. The Serial Driver generates the necessary parity check patterns for the outgoing Command messages and checks them for the incoming Reply messages.

The Serial Driver resides on a single-width 6U VME board. A CAMAC access consists of writing to and reading from certain registers in the Serial Driver as shown in Figure 2. For a CAMAC write, one first loads the Command registers with SC, SN, SA and SF, and then loads the data into the Data Write registers W24-W17 and W16-W1 which generates the Command message. After a certain time which depends on the speed of the Serial Highway (e.g. about 55us in 2.5MHz bit serial systems), the VME processor can read the status (relative address $0C) which, after the Serial Driver having received the Reply message of the addressed Serial Crate Controller, contains the Reply. The Reply can be checked; if no error occurred, the next CAMAC function can be executed.

Reading of the Reply message can be done by polling a status bit NBUSY (which is reset at the start of the Command message and set with the arrival of the Reply message) or by connecting the read to an interrupt generated by the Reply message. If after a certain time (corresponding to 320 bytes in the message stream) the Reply message has not arrived, the NBUSY bit and the interrupt are set together with the corresponding error bits in the status register.

CAMAC read and control operations are generated by loading the command register. After having read the Reply message, a control operation is terminated. For a read operation, the data word has to be read from the data read registers.

The software driver must assure that the sequence of commands necessary for a CAMAC access cannot be interrupted by another CAMAC access to the same Serial Driver.

Arriving Demand messages are stored in a FIFO (first-in-first-out) memory. This assures that all possible LAMs can be served properly. The arrival of a Demand message is signalized by an interrupt.

So, one has 2 internal interrupt sources. They are treated by the VME compatible interrupter chip MC68153. Interrupt levels, interrupt vectors and masking bits are freely programmable. The Reply message generates an INT0, the Demand message an INT1.

<table>
<tr><th>RELATIVE ADDRESS</th><th>D15</th><th></th><th></th><th></th><th></th><th></th><th></th><th>D8</th><th>D7</th><th></th><th></th><th></th><th></th><th></th><th></th><th>D0</th><th>ACCESS</th></tr>
<tr><td>00</td><td colspan="4">SA</td><td></td><td colspan="5">SF</td><td></td><td colspan="5">SN</td><td>R/W</td></tr>
<tr><td>02</td><td colspan="10"></td><td colspan="6">SC</td><td>R/W</td></tr>
<tr><td>04</td><td colspan="10"></td><td colspan="6">W24-W17</td><td>R/W</td></tr>
<tr><td>06</td><td colspan="16">W16-W1</td><td>R/W</td></tr>
<tr><td>08</td><td colspan="10"></td><td colspan="6">R24-R17</td><td>R</td></tr>
<tr><td>0A</td><td colspan="16">R16-R1</td><td>R</td></tr>
<tr><td>0C</td><td>NBUSY</td><td>CERR</td><td>FWE</td><td>ERR CPL</td><td>ERR HED</td><td>ERR CP</td><td>ERR PB</td><td>NO SYNC</td><td colspan="4"></td><td>DERR</td><td>SQ</td><td>SX</td><td>ERR</td><td>R</td></tr>
<tr><td>0E</td><td>ERR DM</td><td></td><td colspan="6">SC</td><td colspan="3"></td><td colspan="5">SGL</td><td>R</td></tr>
<tr><td>10</td><td colspan="8"></td><td colspan="8">CONTROL REG. REPLY INT0</td><td>R/W</td></tr>
<tr><td>12</td><td colspan="8"></td><td colspan="8">CONTROL REG. DEMAND INT1</td><td>R/W</td></tr>
<tr><td>14</td><td colspan="8"></td><td colspan="8">CONTROL REG. INT2 (NOT USED)</td><td>R/W</td></tr>
<tr><td>16</td><td colspan="8"></td><td colspan="8">CONTROL REG. INT3 (NOT USED)</td><td>R/W</td></tr>
<tr><td>18</td><td colspan="8"></td><td colspan="8">VECTOR REG. REPLY INT0</td><td>R/W</td></tr>
<tr><td>1A</td><td colspan="8"></td><td colspan="8">VECTOR REG. DEMAND INT1</td><td>R/W</td></tr>
<tr><td>1C</td><td colspan="8"></td><td colspan="8">VECTOR REG. INT2 (NOT USED)</td><td>R/W</td></tr>
<tr><td>1E</td><td colspan="8"></td><td colspan="8">VECTOR REG. INT3 (NOT USED)</td><td>R/W</td></tr>
<tr><td>20</td><td colspan="8"></td><td colspan="8"></td><td>R</td></tr>
</table>

Figure 2. Address allocation of Serial Driver

The address space needed by the Serial Driver is only 17 word addresses (from $XX00 to $XX20, $ meaning hexadecimal), so short addressing mode has been chosen. The base address is set by a switch in steps of $800 (permitting 32 different address settings). The address modifier codes accepted are $2D (SDVME is only accessible in supervisor mode) or $29 and $2D (accessible in user and supervisor mode), the configuration is selectable by a switch. Figure 2 shows the address allocation of the SDVME and the location of the command and error bits. SA, SF, SN, SC are the serial subaddress, function, station and crate numbers. W1-24 are the write, R1-24 the read registers. Address $XX0C contains the status register.

The meaning of the status bits is as follows. The 4 LS bits (ERR, SX, SQ, DERR) are inserted by the Serial Crate Controller into the Reply message. The other bits are generated by the SDVME:

ERR	Command message was found by the Serial Crate Controller to be non correct,
SX	X-response of CAMAC access,
SQ	Q-response of CAMAC access,
DERR	delayed error, see Chapter 12.2 in EUR 6100e,
NBUSY	Reply message has arrived which means a Command - Reply transaction has been completed,
CERR	error was detected, it is the OR signal of the error bits ERR PB, ERR CP, ERR HED, ERR CPL and NOSYNC,
FNE	FIFO for Demand messages not empty,
ERR CPL	Reply message not complete, i.e. number of bytes wrong,
ERR HED	header in Reply message wrong,
ERR CP	column parity error,
ERR PB	byte parity error,
NOSYNC	synchronization lost, i.e. clock is no longer detected at the input port.

The register at relative address $0E contains the Demand message and an error bit: SGL is the Serial graded LAM pattern, ERR DM indicates an error in the Demand message, i.e. byte parity or column parity error. This register can only be read.

The following 8 registers at $10...$1E serve to program the MC68153 interrupter chip. For the exact meaning of the bits refer to the MC68153 data sheet. Only the first two control and vector registers are used.

The last address (at read access) does a complete reset of the Serial Driver, like SYSRESET on the VME bus.

An exhaustive description of the hardware can be found in [2].

IV. SPECIFICATION SUMMARY

- 6U single VME slave board with P1 connector
- register generated single CAMAC access
- conservative mode and PIO only
- bit and byte serial mode, 5, 2.5, 1 and 0.5 MHz selectable
- appropriate insertion of space bytes
- stacking of Serial Demand messages in a FIFO
- Reply message generates maskable interrupt, polling possible
- Demand message generates maskable interrupt, polling possible
- VME characteristic A16/ D16, I(x), x programmable
- low power consumption (6.5W) due to utilization of CMOS

V. DRIVER SOFTWARE

The driver for single CAMAC instructions and block transfer runs under LYNX OS and is written in C. Originally, it was written under OS-9, but then transported to the new PS standard operating system which is the POSIX compliant operating system LYNX OS. The execution speed per CAMAC instruction for a list of CAMAC functions is 100Kwords/s (for 24 or 16 bit words). The speed for block transfer (executing the same CAMAC read or write function) is 200Kwords/s. Both values hold for 5Mbyte/s transmission speed on the Serial Highway and a 68030 CPU running at 25MHz. For 2.5Mbit/s transmission speed, 45us have to be added per word transfer.

VI. ACKNOWLEDGMENTS

The original design of the SDVME was done by L. Antonov and V. Dimitrov, CLANP, Sofia. The software driver under LYNX OS was written by A. Gagnaire, CERN.

VII. REFERENCES

[1] A. Gagnaire, N. de Metz-Noblat, Ch. Serre, C.H. Sicard, "Replacing PS Controls front end minicomputers by VME based 32-bit processors", November 1991, these Proceedings, (ICALEPCS, Tsukuba, Japan, 1991).

[2] L. Antonov, V. Dimitrov, W. Heinze, "A CAMAC Serial Highway Driver in VME", PS/CO/Note 91- 22, 1991.

Fast Automatic System for Measurements of Beam Parameters of the MMF Linac

P. Reinhardt-Nickulin, S. Bragin
N. Ilinsky, Yu. Senichev
Institute for Nuclear Research
Moscow, USSR

Abstract

Fast transverse beam profile and current monitoring systems have been tested at the Linear Accelerator of Moscow Meson Factory. The signals for each system are derived from multiwire secondary emission chamber and beam current transformer. Each beam pulse is digitized by fast ADC's. There are two modes for systems. First one is for detailed beam adjustment and second one is for normal 100 Hz rate of the MMF Linac. Essential features of the hardware, software, data acquisition, measurement accuracy and beam results are presented.

1 Introduction

Systems for automatized measurement of beam parameters are components of general control systems of accelerators. Just through computerized measurement systems feedbacks enveloping accelerator completely or partially are closed [1]. Application of measurement systems permits first of all to make effective tuning of beam parameters such as transverse sizes, emittance and so forth, and, secondly (and it may be the most important) to reduce effects of radiation induced by the beam. This makes essentially easier exploitation conditions and conditions of tuning works, because for getting needful information about the beam it is necessary to spend essentially lesser beam time. Supposed installation in spaces between resonators of the first part of the MMF Linac measuring assemblies, consisting of multiwire electrodes, phase analyzer and target for energy estimation must solve tuning problems to a considerable extent.

In this paper it is considered principles of construction and main features of data treating systems, which permit to get as simple transverse profile of each whole beam pulse in usual exploitation regime as detailed picture of evolution of transverse profile and intensity for single beam pulse, and on the base of this information to estimate mutual influence of neutralization process and Coulomb's repulsion. Besides that high registration speed of profiles and intensity gives the base for hope on getting more full information needful for tuning of accelerator resonators by the beam.

Signals from multiwire chamber [2] and current transformer (Fig. 1) are treated through specialized set of modules and standard CAMAC modules. This set forms flexible enough complex of equipment, organizing the work as with objects placed near computer as with remote ones.

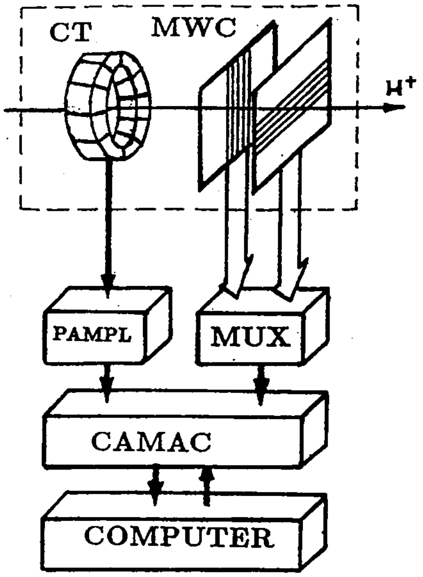


Figure 1: Simplified block diagram of beam parameter measurement system.

2 Speed criterion

Taking into account, that there will be approximately 650 analog signals from the wires placed in proton transfer channel (transfer channel is placed between injector and first accelerating resonator) and in spaces between resonators, it is easy to see, that detailed digitizing of these data by means of separate ADC for every wire is not simple and very expensive way. Therefore construction of system should be made by traditional way of storing and multiplexing of analog data. And criterion of information detailing one must search being attached not only to speed of ADC and computer, but mainly to physical processes and their characteristic frequencies, which may be studied, not converting treating system into too expensive one. In this case the most interesting tasks are ones bound up as mentioned above with influence of space charge neutralization, and also with inertness of resonator feedbacks. During beam injection it takes place ionization of residual gas in transfer channel, that can create at the beginning of beam pulse distinctive change of the form of the envelope because of change of the phase shift of radial oscillations. It can lead to unforeseen losses and to current limitation because of mismatching. Because neutralization time takes up $5 \ldots 10\mu s$ at designed vacuum (and greatly depends on vacuum), then this process can be observed only by fast enough system. As the system "resonator-feedback" has crossover frequency of 160 kHz, then for observing influence of this loop on the beam, diagnostic system must be a few times faster. In exploitation when tuning problems are solved the speed of measurements can be equal or less than 100 Hz repetition rate of the Linac.

3 Organization of measurement system

Proposed system consists of three parts :

- *Simple Profile Measurement system* (SPM), which permits to get profile of each pulse of the Linac by means of integration of the currents from the wires of Multiwire Chamber (MWC).
- *Beam Current Transformer system* (BCT), which permits to measure pulse current for each pulse of the Linac.
- *Fast Beam Parameter Measurement system* (FBPM), which permits to get series of beam profiles measured within one beam pulse at tuning Linac regime or permits to make SPM system measurement at normal 100 Hz Linac exploitation.

3.1 Simple Profile Measurement System

SPM hardware consists of analog signal multiplexer (MUX) which is installed near MWC at the Linac, and ADC in CAMAC crate, which is placed in accelerator control room. Methods of multiplexer construction are well-known [3,4] so we limit ourselves only to brief describing of its working. Simplified block diagram of device is given in Fig. 2 . Besides of input RC-circuits multiplexer consists of following parts: analog switches, control logical scheme, circuits of signal forming for switch control, output amplifier. Voltages accumulated on input capacitors are switched in turn to final amplifier and through cable line are fed to ADC. MUX logical scheme is started by external sync pulse SP coming to input of time delay generator (DG) which is needed for shifting of scanning beginning for a time sufficient for elimination of high-frequency inducings arising from high-power pulse equipment of accelerator. Generator of scanning envelope (EG) is started by decay of DG pulse and permits working of generator of scanning train (TG) . Pulses of TG are fed to counter Cnt1. Highest output of Cnt1 is connected to input of counter Cnt2. Cnt1 output code controls working of decoders Dc1 - Dc4, and Cnt2 code controls working of Dc5 which permits working of Dc1 - Dc4 in turn. Signals from Dc1 - Dc4 outputs converted by forming circuits (FC) open switches S1 - S62 in turn. Switch S63 is opened between scanning pulses and prevents charge accumulation on input capacitance of final amplifier. For determination of profile for each beam pulse there is unit in scheme for repeated start of system and discharge of capacitors. It consists of generator of envelope repetition (ERG) and switch S64 connected to final amplifier input. After closing the switch S65 every decay of EG pulse starts ERG, which opens S64 and permits to begin repeated cycle of scanning. Thus it is carried out in turn discharge of all capacitors through switches S1 - S62 and switch S64.

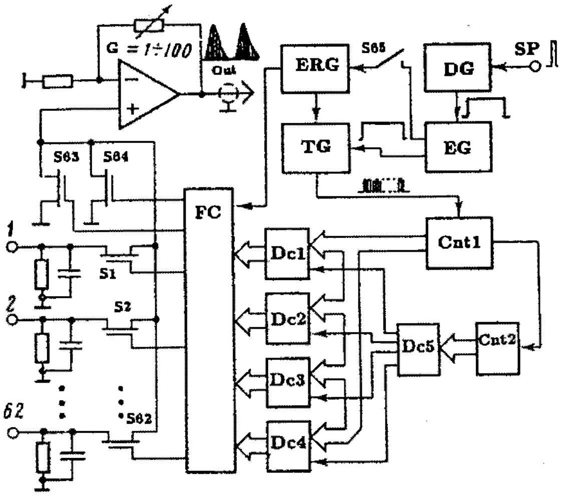


Figure 2: Block diagram of simple MUX.

One of the advantages of computer treating is subtraction of voltages existing on the capacitors when no beam passes through MWC. Storing and subtracting of low-frequency inducings permit us almost completely to eliminate their influence on measurement results.

3.2 Beam Current Transformer System

BCT system consists of low noise preamplifier (PAMPL) installed near ferrite current transformer (CT) at accelerator, main amplifier with scheme for suppression of low-frequency inducings and ADC installed in CAMAC crate in accelerator control room. Working out amplifying tract of CT we paid attention to decreasing of level of circuit noises and external inducings [5]. In system "CT - amplifying tract" it was carried out optimization based on the criterion of signal/noise (S/N) maximum ratio. As result of this investigation it was found maximum S/N at number of turns of CT equal 170. The level of noise corresponds 80 nA of beam current. Existing software for BCT system is directed to subtraction of background, amplitude analysis of current pulses and getting some quantitative information for amplitude spectra (histograms).

3.3 Fast Beam Parameter Measurement System

FBPM system can measure a series of proton beam profiles with time interval between adjacent profiles equal 1, 4μs and more. The measurements of beam current can be done within the same time interval by BCT system with separate fast ADC. Simplified scheme of system is given in Fig. 3 and consists of two main parts:

- Head part, i.e. complex of equipment placed near accelerator;
- Equipment placed in control room of accelerator.

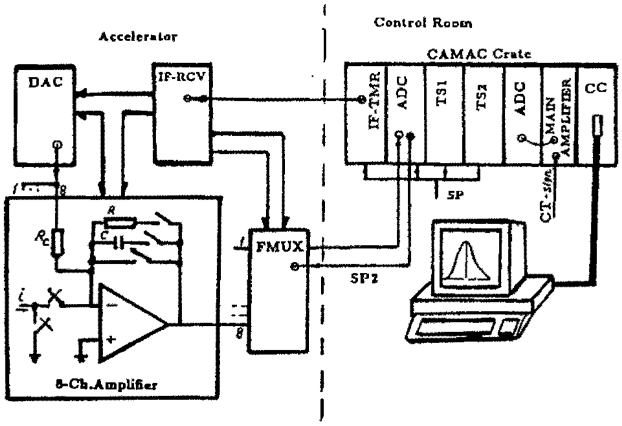


Figure 3: Simplified scheme of fast beam parameter measurement system.

Let's consider the structure and work of the main elements of the head part. Signals from MWC pass through 8-channel matching amplifier with switched gain factor and mode of work. In tuning mode amplifier works as linear amplifier, in exploitation mode it works as charge sensitive amplifier with gated input. In tuning mode signals enter the fast multiplexer (FMUX) which selects and holds time samples of beam profiles and takes out them in consecutive order to the cable leading to fast ADC. ADC – FMUX synchronization is provided by series of pulses (SP2) from ADC. Besides these modules head part contains also DAC and interface-receiver (IF-RCV), which are intended for synchronization, calibration and switching of the work mode.

Part of the system placed in control room of accelerator, consists of microcomputer and five modules (fast ADC, two timer-synchronizators (TS1 and TS2), interface-transmitter (IF-TMR) and crate-controller (CC)), placed in CAMAC crate. Module ADC has digital memory. This module makes digitizing of signals and then memorizes received information about the magnitudes of signals for subsequent reading them into computer memory. Module TS1 is intended for time attaching of the beginning of program running to synchronization pulse (SP) of accelerator. Let's consider structure and work of the part of complex which permits to carry out calibration and linearization of signal transmission tracts. This part of complex includes DAC, two interfaces (IF-TMR and IF-RCV) and TS2. The base of calibration algorithm consists in feeding of rectangular pulse from DAC to all inputs of matching amplifier. As DAC should be placed near matching amplifier, then transmission of digital code from computer to DAC is organized by modules IF-TMR and IF-RCV. Module IF-TMR provides recoding of parallel code of calibration pulse amplitude to serial code for transmission through the cable. Interface IF-RCV makes inverse transformation. Besides that module IF-TMR transmits special code words needful for time attaching of calibration pulses to multiplexer synchronization. Main stage of calibration is carried out when there are no beam current signals on MWC (for example, when MWC is taken out of beam) and consists in making up the work table of correspondence between DAC codes and codes being read from ADC memory. For this purpose it is set consecutively a few tens of DAC amplitudes with equal intervals between them. When measuring profile, inverse transformation is carried out. This permits to decrease considerably influence of following factors on final results of measurements:

1) spread of transmission coefficients of the channels;
2) non-linearity of tracts;
3) constant low-frequency and impulse inducings.

Besides that system permits to control any change of these factors directly in process of measurement. With this purpose in module IF-TMR it is organized starting of second sync pulse delayed from SP by module TS2 for the time sufficient for reading of all information from ADC memory to computer; and to DAC it is transmitted code of amplitude of verifying pulse and command for its forming. Thus, it is fed additional verifying pulse of known amplitude to calibration input of matching amplifier in every period between beam pulses, and it is started multiplexer, ADC and program for reading of response information to the computer memory. Result is compared with table (created at the beginning of measurement session with using of second sync pulse) and then is led out to videoterminal screen. It permits operator to decide whether it is needed repetition of the measurement because of some accidental momentary noise influence on the system (or maybe carrying out of new cycle of system calibration because of changed conditions of system work). If we are satisfied by test results then we convert, in accordance with work table, output values of ADC to corresponding input values of DAC. Because table is got only in finite number of points, then we need linear interpolation. After converting the array of output ADC values to equivalent DAC values we normalize result array on maximum value and lead out information in form of profiles. Then we either repeat cycle of measurements or go out of program.

Results of bench test measurements

System was tested on the stand. To all 8 inputs of system it was fed the same rectangular pulse. Following tests were carried out:

1) *Comparison of profiles which were got without calibration table and with table.* In both cases it was fed signal equal the half of maximum input signal. Without table we had spread of profile top of 20...30%, then with table 2...3%.

2) *Measurement at different speed of counting.* That is to say it was carried out measurements when varying the scanning time between multiplexer channels from some t_0 to t_{min}. It was got amplitude spectra (histograms) of output signal at different speed of counting. Results showed, that decreasing of scanning time less than 175 ns, which corresponds to the time of writing of one profile 1.4μs , gives rise of equipment errors because of limited speed of writing to ADC memory.

3) *Temperature drift.* It was received amplitude spectra of output signal in some time interval of system running. These measurements indicate that at room temperature one can work with the system without re-calibration during several hours not distorting measurement results.

4) *System linearity.* Research of linearity indicates that distortions are less than ±1%.

5) *Measurements with slow and fast inducings.* It was carried out measurements in conditions when slow inducing makes up approximately 8 % of signal amplitude, and fast local inducing makes up about 50 % of signal amplitude. Note, that given inducings were stable in time. Under these conditions due to calibration with the same strays we could avoid their influence and achieve rectangular profile with spread of top about 2...3%.

FBPM system is preparing now for installation on the accelerator with new measurement assembly.

4 Beam results

Simple profile measurement system and beam current transformer system were tested on proton beam transfer channel of the MMF Linac between injector and first accelerating resonator. MWC was installed perpendicularly to beam axis. Multiplexer was placed at the distance of ≈ 1 m from transfer channel. Information about shape and amplitude of pulse beam current was got with the help of our current transformer and BCT system. Serviceability of system was tested by introducing of collimator into channel at the distance of ≈ 0.8 m from CT and MWC. Changes of profiles after introducing of collimator (15 mm) is shown in Fig. 4 . Introducing of collimator decreases as signal amplitude as profile width. At pulse proton current ≈ 120 mA, pulse duration ≈ 70μs the amplitude of signal on central wire was a few Volts and inducing signals were not more than 2 ADC channels (20 mV). Stability of beam current was measured by BCT. Pulse current shapes that were got by BCT system had more details than ones from usual current transformer monitor at the transfer channel. The inducings, that were measured in BCT tract, had maximum value 0.2% of usual pulse current magnitude in transfer channel.

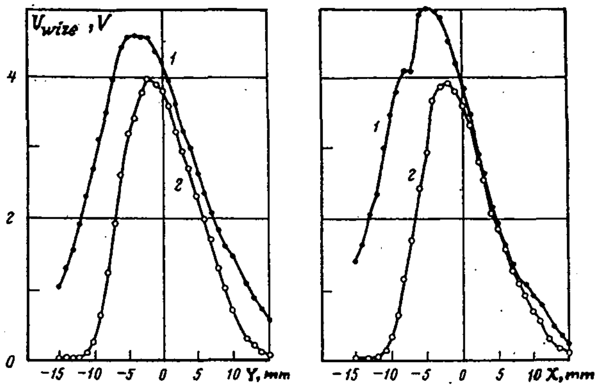


Figure 4: Beam profile change after introducing of collimator: 1)without, 2)with collimator.

References

[1] E.J. Soderstrom et al., Proceedings of 1989 IEEE Particle Accelerator Conference, March 20-23, 1989, Chicago, 89CH2669-0, v.3 of 3, p.1864.

[2] S. Bragin, N. Ilinsky, P. Reinhardt-Nickulin, Yu. Senichev, Prib. Tekh. Eksp., No.3, 1990, p.59-62 (in Russian).

[3] S. Bragin et al., Prib. Tekh. Eksp., No.6, 1990, p.76-79 (in Russian).

[4] Lee D.M., Brown D., Hardekopf R. et al., IEEE Trans. on Nucl. Sci. 1985, v. NS-32, No.5, p.1967.

[5] Gallegos F.R., Morrison L.J., Browman A.A., IEEE Trans. on Nucl. Sci., 1985, v. NS-32, No.5, p.1959.

Beam Position Monitor Multiplexer Controller Upgrade at the LAMPF Proton Storage Ring*

W. K. Scarborough and S. Cohen
Los Alamos National Laboratory
Los Alamos, NM USA 87545

Abstract

The beam position monitor (BPM) is one of the primary diagnostic tools used for the tuning of the proton storage ring (PSR) at the Clinton P. Anderson Meson Physics Facility (LAMPF). A replacement for the existing, monolithic, wire-wrapped microprocessor-based BPM multiplexer controller has been built. The controller has been redesigned as a modular system retaining the same functionality of the original system built in 1981. Individual printed circuit cards are used for each controller function to insure greater maintainability and ease of keeping a spare parts inventory. Programmable logic device technology has substantially reduced the component count of the new controller. Diagnostic software was written to support the development of the upgraded controller. The new software actually uncovered some flaws in the original CAMAC interface.

I INTRODUCTION

The Beam Position Monitor (BPM) system is the primary tool available for beam tuning at the Clinton P. Anderson Meson Physics Facility (LAMPF) Proton Storage Ring (PSR). The BPM multiplexer controller is an integral part of the BPM system. The multiplexer controller is the interface between the BPM[1] system hardware and the PSR microVAX data acquisition and control system, see Figure 1. There are approximately seventy BPM's that are used as a primary tuning tool by PSR operators.

The existing multiplexer controller was difficult to troubleshoot and repair. The old controller was built on 12 wire-wrap "CASH" cards mounted in a 19 inch rack mount chassis. A spare controller chassis was never built, making on line troubleshooting to the component level necessary.

The analog to digital converters that were used in the original design are no longer available. It should be noted that conversion must occur with in the time constant of the circulating proton bunch in the ring, 360 ns.

In addition, there were no software diagnostic tools to aid in troubleshooting and testing.

Several factors affected design of the new controller. Programmer resources were not available to write code for a more modern microprocessor so we needed to do an Intel 8085-based design and reuse as much of the 3000 lines of original assembly code as possible. The new controller needed to be modular so that "card swapping" could be used as a troubleshooting and repair technique. We wanted to use as many off the shelf components as possible. The new controller needed to be as "plug compatible" with the old controller as possible, so that no modifications to other components of the multiplexer system were necessary.

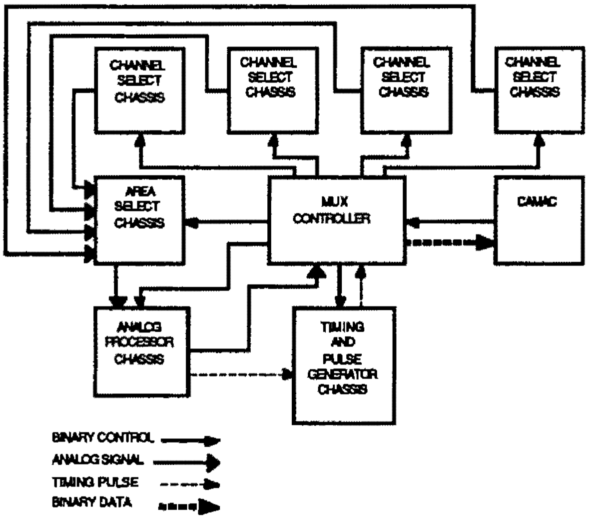


Figure 1. BPM System Block Diagram

We elected to do the project in a STD bus format. STD bus is a mature, well defined, well supported industry standard. The STD bus standard was designed for 8-bit microprocessor control functions, and several vendors offer 8085 CPU cards. In addition, several vendors offer well-built chassis with terminated back planes.

II. CONTROLLER FUNCTIONS

When the BPM control software running on PSR microVAX needs data, it sends a command/request list via CAMAC, to the controller. The request defines which channels are to be read, number of samples per channel, and timing information. The command list is loaded into a CAMAC output FIFO(First In First Out buffer) and then signals the multiplexer controller, via a CAMAC TTL output module, that the FIFO is loaded and ready to be read. The controller reads the FIFO, stores all the command/request information in RAM.

The controller takes a number of actions prior to actual data acquisition. It must first select the channel and area multiplexer that corresponds to the BPM whose position information has been requested. The controller writes to a timing chassis to set up trigger and timing parameters and it writes to the analog signal processor chassis to set gain and

---

*Work supported by the US Department of Energy

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S11LLC06

attenuation of the BPM front-end analog processor. Finally, its internal counters are initialized to begin counting channels and samples per channel. After the set up is complete the controller accepts analog signals and begins analog to digital conversion.

When the controller has acquired all the requested beam position information, it formats the data, flags any known bad data, and loads it into a CAMAC input FIFO. Then the controller pulses a CAMAC input register that sets an interrupt for reading the BPM data that has been loaded into the input FIFO.

III. CIRCUIT DESCRIPTION

A. CAMAC Interface Circuit Board

The CAMAC interface card is the interface between the controller and the CAMAC crate connected to the PSR microVAX via a CAMAC serial highway.

B. I/O Functions Circuit Board

The controller talks to the Area Select chassis, the several Channel Select chassis, and the Timing and Gate Generator chassis via the three I/0 cards. All three cards are similar in that each card decodes I/0 write instructions and latches the data bus into buffers. The differences are that the port addresses differ from card to card.

The instruction decoding, on all the I/0 cards, occurs in programmable logic device (PLD) decode chip.

C. Timing and Control Circuit Board

The Timing and Control card generates the timing pulses that other cards use to initiate analog to digital conversions, strobe data in to RAM and increment counters.

D. Analog to Digital Conversion Circuit Board

The three A/D cards, one each for horizontal position, vertical position and beam intensity are identical except for the address decoding jumpers.

F. Beam Status Module & Counter Module

The Beam Status card uses one bit of a 2K RAM to store whether or not beam was present at the time of an analog to digital conversions.

The microprocessor subroutine that packs the data for shipping to the VAX looks at the beam status data and flags the horizontal and vertical position data if beam was not present. The bad data flag is an 80h[2] in the position byte. When the VAX software sees the 80h in a stream of data it ignores that byte and does not attempt to display it.

The main function of the counter card is to count Samples Per Channel (SPC).

G. CPU Circuit Board

The CPU is a commercial STD board available from Microlink, a division of Sea-Ilan, Inc.[3] .

We have made several changes to the CPU card to accommodate non-standard STD Bus signals that are routed on the STD backplane.

The 8085 is designed to boot to address 0000h however the ROM on the CPU circuit board resides at 2000h. We have included a special boot sequence to take care of this.

H. Chassis/Backplane Circuit Board

The multiplexer controller resides in a Matrix Corporation "Blackplane" STD bus card cage. The chassis was purchased assembled with back plane and fused and switched power supply. The power supply is rated at 5 volts, 25 amps and +/- 12 volts, 3 amps.

IV. SOFTWARE

Of the 3000 lines of 8085 assembly code that is the heart of the controller, only about 20 lines needed to be changed. Most of the changes were in response to the hardware-specific boot sequence required by the new CPU card. The mapping of memory needed to be modified in order to accommodate the address space available on the commercial STD cards.

A PC-based (MS-DOS) diagnostic was written to assist in the development of the new controller. The program was written in C with a modifications to the software library supplied with the CAMAC crate controller used for our CAMAC-multiplexer testbed so that software interface to the library routines was the same as those on the PSR microVAX. The source code for the diagnostic program was easily ported to the VAX with very few changes.

The VAX-based diagnostic can be run from any VT-type terminal. One can modify the command list which is sent to the multiplexer controller. By modifying the command list one can control channel selection, time into pulse, time into cycle, trigger selection, sensitivity, and samples per channel. The diagnostics will send the list to the controller, handling all the handshaking just as the production software used by the operators does. When the controller has acquired the data it saved to disk,where it is available for viewing and analysis.

The diagnostic program uncovered a CAMAC interface timing problem, that existed with the old controller, which had the potential to cause data to be lost.

V. RUN-TIME EXPERIENCE

The controller has been used for about four months of PSR production. We have had no major problems with the system. The addition of needed diagnostic software has allowed us to accurately pinpoint the sources of problems that have come up.

---

[1] E.F. Higgins and F.D. Wells, " A Beam Position Monitor System for the Proton Storage Ring and LAMPF", Particle Accelerator Conference, Washington, DC USA (1981)

[2]The lower-case "h" indicates a hexidecimal number.

[3] "STD-245, 8085A Single Board Computer, Operations Manual" Microlink, a division of Sea-Ilan, Inc.,14602 N. US Hwy 31, Carmel, IN USA 46032

The KEK PS Fast Beam Loss Monitor System

J.A. Holt*, J. Kishiro, D. Arakawa, H. Someya and S. Hiramatsu
National laboratory for High Energy Physics, KEK
1-1, Oho, Tsukuba, Ibaraki, 305, Japan

Abstract

The higher beam intensities now being accelerated in the KEK proton synchrotron (PS) complex have increased the importance of observing the beam loss during acceleration. The beam loss should be continuously monitored to minimize radiation damage to the accelerator components. A fast loss monitor also is a good tool for observing where and when the beam is lost, by which we are able to get information on the beam dynamics. The development of a fast beam loss monitor system at KEK is described in this paper.

I. INTRODUCTION

The beam intensity in the KEK PS has gradually increased and the PS continuously is operated with an intensity of about $3x10^{12}$ppp. There are, however, some problems in maintaining this intensity in the accelerator and in the beam transport lines. The beam loss might come not only from a miss steering of the beam orbit but also from the short time scale dynamical behavior of the beam bunches. One property of a loss monitor which should be noted is the extremely high signal to noise ratio. In the usual beam monitor systems, the signal is proportional to all of the particles in the bunch. Any loss causing perturbation signal due to some short time behavior of the particles has to be extracted from the signal fluctuation. In a loss monitor, only the perturbed signal is seen.

We have adopted at KEK a secondary electron multiplier vacuum tube as a beam loss detector because of its good time response, compactness and ease of handling. The time response was tested in the TRISTAN electron-positron storage ring which has a bunch length of 300ps. The tube response was quite good, about 40 ns (Fig 1). This time response is good enough to enable turn by turn beam loss monitoring of an individual bunch in the proton synchrotron complex.

As an initial test, thirteen detectors were distributed around the accelerator complex. The PS complex consists of a 750 KeV Cockcroft-Walton, a 40 MeV linac, a 500 MeV Booster and the 12 GeV main ring. There is a transport line between the linac and booster and another line between the booster and main ring. One detector was placed near the 40 MeV transport line, eight detectors were placed around the booster, one detector near the 500 MeV transport line and two detectors in the main ring. The detected signals were digitized by fast CAMAC ADCs and acquired by a VME computer for analysis and display. The control and display software was written using X-windows under UNIX. This enabled simultaneous display of multiple detector signals and the ability to display the data on any X-terminal on the network.

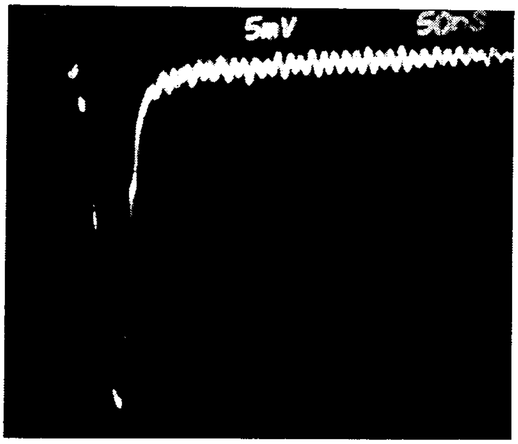


Figure 1. Time response of the detector.

II. HARDWARE CONFIGURATION

A. Detector

The detector is a secondary electron multiplier tube R595 made by Hamamatsu. The gain of the detector is around 10^5 - 10^7. Since the PS loss rate is very high, the tube gain is more than enough. The tube is installed into an aluminum case for light and noise shielding. Attached to the back of

---

[*] Presently at FNAL, Batavia, IL, U.S.A

the case is a 20dB gain amplifier to avoid signal to noise ratio degradation from the detector to the electronics building located outside the tunnel (Fig. 2).

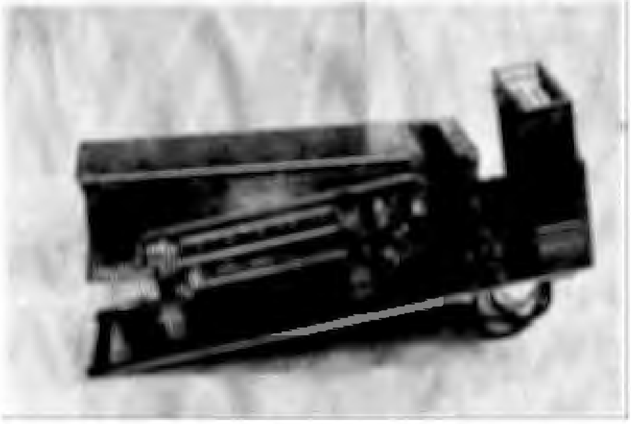


Figure 2 An overview of the detector.

The detector case is attached to the tunnel wall pointing to the beam pipe of interest. The front wall of the case is carefully adjusted so as to degrade the lower energy component of the background residual radiation. The gain calibration of each tube is still in progress. There is not enough data available yet to check performance degradation due to radiation damage of the tube.

B. CAMAC system

The detector signals are digitized by CAMAC ADCs located in a building outside of the accelerator tunnel. All timing signals and gating are controlled by CAMAC modules. For the beam transport lines a 100 MHz ADC is used. The booster accelerates one bunch in 25 msec. To acquire the turn by turn beam loss with sufficient resolution, a 100 MHz ADC with 2.5 MBytes of memory is used (Fig. 3).

Since the main ring accelerates nine bunches, a bunch selection system is used to select the bunch of interest. Also since the acceleration ramp is rather long, data along the whole ramp cannot be acquired. Instead only the loss

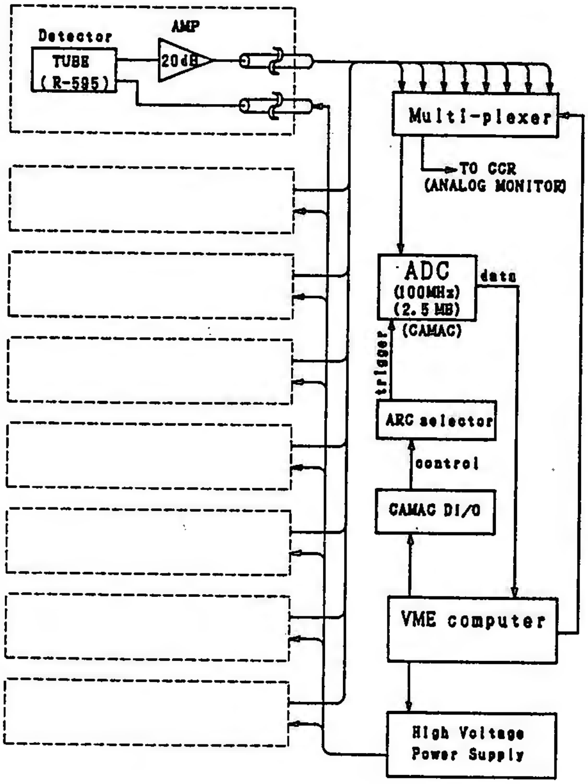


Figure 3 Booster loss system configuration.

signal peak is digitized using a specially developed ADC and stored in memory. Figure 4 shows the nine bunch beam loss in the main ring. All ADCs have hardware memory so that the computer is not used during the actual acquiring of data (Fig. 5).

The ADC system described above enables us to observe time dependent beam loss for an individual bunch throughout the entire acceleration cycle. Beam loss due to betatron oscillation for instance, can be observed which will help in tuning the ring correctly.

The CAMAC crates are connected to a VME computer via the Branch Highway interface. The Branch Highway to VME interface card has hardware interrupt inputs which are used to signal the end of the acceleration cycle.

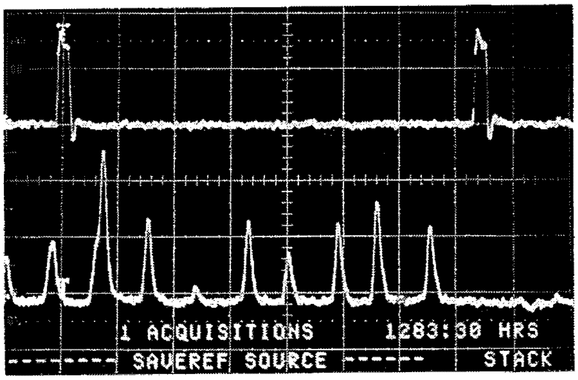


Figure 4 Nine bunch loss in the 12GeV main ring.

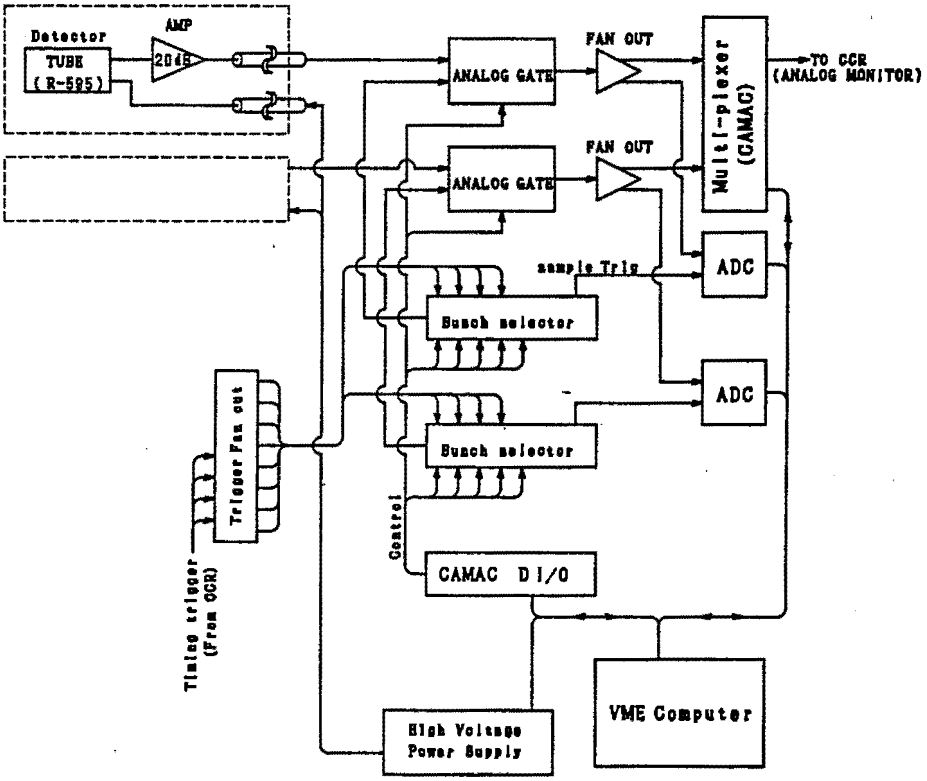


Figure 5 Main ring loss system configuration.

III. SOFTWARE

The data acquisition system software consists of a number of separate application programs written in C using X-windows and the OSF/Motif toolkit. The operating system is UNIX.

The CAMAC hardware information is stored in a database. Using a menu-driven program the user is able to easily update the location and module types, the command list, etc. of the CAMAC hardware in the system. The data acquisition program reads this database at start up as its only source of hardware information thus requiring no software changes when there is a hardware change.

The actual data acquisition is made up of several processes running simultaneously. There is a control process, a CAMAC process, and a separate graph process for each detector signal being acquired. Except for the signal data to be displayed all interprocess communication is handled by custom X events. The signal data to be graphed is passed via the UNIX message facility.

The control process oversees all aspects of the data acquisition system. This process maintains an additional database which lists which CAMAC commands start or stop the data acquisition for each detector, commands to read in the data, control of the detector voltage, and timing control. The user can create several software "detectors" for each hardware detector corresponding to different types of data desired.

The CAMAC process is the only means of communication with the hardware using a UNIX device driver developed especially for this purpose. This device driver has a command list mode in which the CAMAC instructions for all detectors are concatenated into a single list by the control process, passed to the CAMAC process and loaded into the device driver. At a hardware interrupt indicating the acceleration cycle has completed, this instruction list is executed and the data is passed back to the CAMAC process. The CAMAC process scales the data appropriately and sends the scaled data to the proper graph process by a UNIX message.

A separate graph process is running for each detector signal being acquired. The graph process sends to the CAMAC process graph scale and window size information so that the data can be scaled properly. The data is scaled in the CAMAC process to minimize the amount of data which has to be passed between processes; all redundant data are thrown away.

Although an object oriented language was not used for software development, the data structures and functions were constructed in an object oriented fashion. As a result there is good isolation between different types of data structures and functions which makes extension and debugging of the system easier. Also there are no hard-coded assumptions about the type and number of detectors so that this system could be used for other purposes.

IV. SYSTEM PERFORMANCE

Figure 6 shows a typical display. Multiple detector windows as well as the control window are shown. The bunch tagging system is very effective in acquiring data from the same bunch throughout the accelerator complex. A comparison between the data displayed in several windows clearly shows where and when the contents of the bunch is lost. The beam loss shown in a single window varies from cycle to cycle showing the lack of beam stability. In the beam loss data for the booster loss has been observed outside of the rf bucket during acceleration. This type of loss cannot be observed very well with the usual type of beam monitors because this signal is buried in the signal from the particles in the rf bucket.

Although the system is working a number of improvements need to be made. Due to the large amount of data acquired (3 Mbytes), it takes two acceleration cycles (one cycle = 5 sec) for the graph windows to update. Considerable speed increases can be obtained by optimizing the data scaling routines. Also there are still problems in communication between the CAMAC device driver and the CAMAC process.

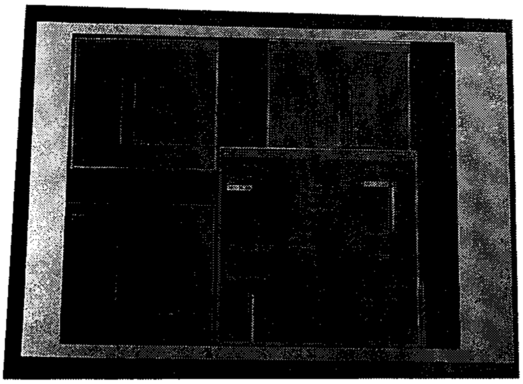


Figure 6 An example of the display picture.

V. CONCLUSION

A development of a fast beam loss system is now successfully going on at KEK. In spite of waiting for some improvements, the turn by turn beam loss observation in the accelerator has been certainly realized. Some examination on radiation damage of the detector and on the gain variation are waiting for soon.

NON-DESTRUCTIVE FAST DATA TAKING SYSTEM OF BEAM PROFILE AND MOMENTUM SPREAD IN KEK-PS

T.KAWAKUBO, E.KADOKURA, T.ISHIDA* Y.AJIMA AND T.ADACHI
National Laboratory for High Energy Physics, 1-1, Oho, Tsukuba-shi, Ibaraki-ken, 305, Japan
*Mitsubishi Electric Company, 1-7-4, Iwamoto-cho, Chiyoda-ku, Tokyo-to, 101, Japan

Abstract

A mountain view of beam profiles in a synchrotron ring can be taken without any beam destruction by collecting charged particles produced by the circulating beam hitting residual gas in the ring to a sensor. When a rectangular Micro Channel Plate with multi-anodes or lined-up electron multipliers is used as the sensor, the profiles can be measured within one acceleration period, even if the beam intensity is very low and the ring is kept in a high vacuum. We describe this non-destructive profile monitor (NDPM) as well as the momentum spread measurement system by a combination of two sets of NDPM.

1. INTRODUCTION

It is very convenient for beam studies and machine operation to measure the beam profile in a synchrotron ring without causing any damage to the circulating beam. The principal of the non-destructive beam profile monitor (NDPM) is to measure the position dependence of the positive ion current produced by the circulating proton beam in a synchrotron ring. Since the current signal is very low, we usually use an element to amplify the signal, such as a micro-channel plate (MCP) or an electron multiplier (EM). We had installed two sets of NDPM by using a large rectangular area MCP with 32 anodes[1), one of which measures the horizontal beam profile in the Booster ring and another in the Main ring. For setting NDPM at a ring position with a large beam size, using an assembly of many EMs as a sensor, is better than using a MCP, from the view point of long life against radiation and a high saturating signal current[2). A combination of two NDPMs (one of which is made of EMs and set at the place with a large dispersion function; the another is made of MCP and set at a location with a small dispersion function) is used for measurements of the momentum spread*. We introduce this measurement result in the Main ring.

The VME computer system takes data from those sensors via an A/D converter, rearranges them and displays a "mountain view" of the transversal beam profiles as well as the time dependence of the beam (center, size, momentum spread) within one acceleration period.

2. NON-DESTRUCTIVE PROFILE MONITOR SYSTEM AND DATA-TAKING METHOD

A.Mechanism and electric circuit

A circulating beam in a synchrotron strikes residual molecules in the vacuum ring while producing positive ions and electron pairs with some probability. When a positive collecting voltage is supplied to an electrode (as shown in Figure 1), positive ions move from the bottom to the top along the collecting field. If a large-area rectangular MCP with multi-anodes or lined-up EMs are placed at the end of the field, they can measure the number of ions which are produced in proportion to the beam intensity along the vertical collecting field. In our case, the MCP is a tandem-type and has an effective area of 81mm×31mm; 32 anodes (each anode has a width of 1.5mm, a length of 29mm and a pitch of 2.5mm) are placed closed to the output side of the surface of the MCP. An EM-type NDPM has 30 lined-up EMs, in which every EM has an aperture with a width of 5.2mm and a length of 30mm. Every anode of the MCP or EM has an independent electric circuit (shown in Figure 2).

---

*The authors would like to acknowledge Dr.K.Narushima, Mr. T.Kubo and Mr.Y.Satoh for helping us to install NDPMs in the vacuum chamber.

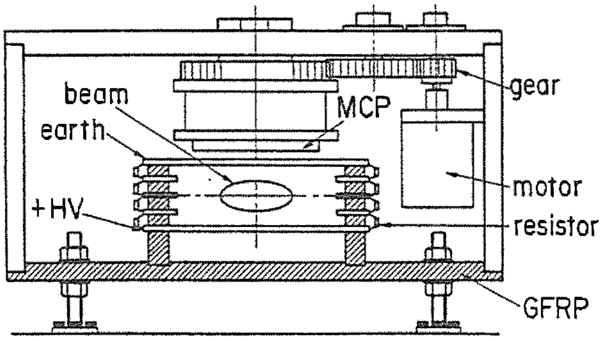


Figure 1. Fundamental plan of a horizontal NDPM with multi-anodes.

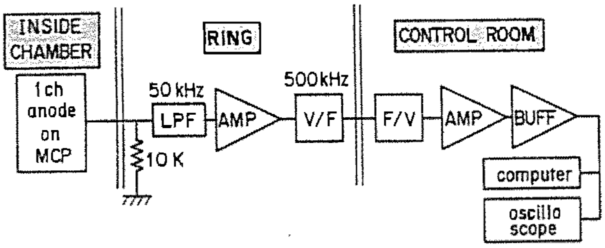


Figure 2. Block diagram of the electric circuit connecting every anode on an MCP or EM.

B. Data taking

Every anode's output signal from beam injection to extraction is memorized by a VME local computer via an A/D converter. The A/D converter has a data-sampling period which can be changed from 3μsec to 255μsec, and eight input terminals. As shown in Figure 3, (in the case of the minimum data-sampling period) the signal from one anode is chosen by a multiplexer for 3μsec with every 24μsec(=3μsec×8ch). Therefore, our NDPM system (which has 32 anodes) requires four A/D converters and the minimum data taking period for one anode is 24μsec. As the data-transfer time from the local computer to the center one is not fast, we set the maximum data-taking number from one anode to 256. When the measuring time range is long, such as the Main ring acceleration period (about 3 sec), we increase the sampling time from the minimum value (=3μsec) and take 256 data by averaging over several data for an increasing S/N ratio.

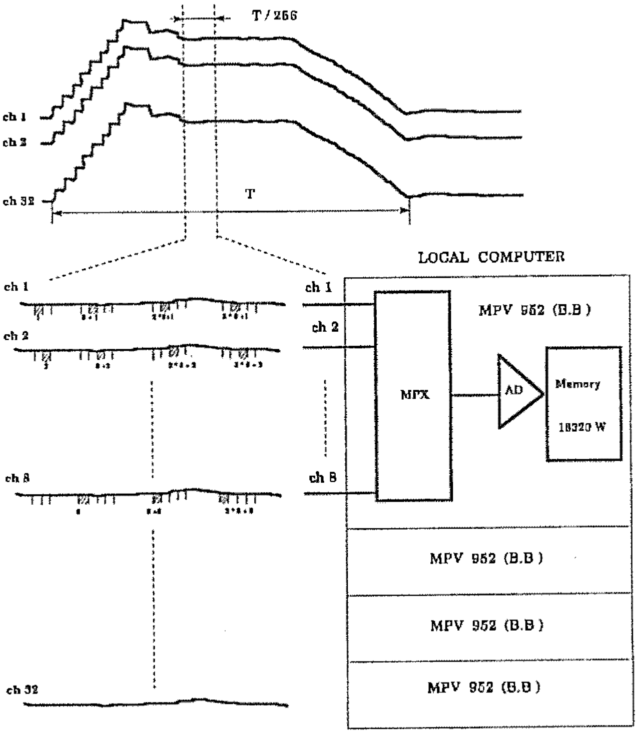


Figure 3. Multiplexer and A/D converter system to take signals from 32 anodes.

C. Calibration

In this system the most important point concerns the calibration of the entire gain, which includes the MCP, pri-amp, voltage-to-frequency converter (VFC), frequency-to-voltage converter (FVC), main-amp and analog-to-digital converter (ADC). For observations of the beam profile, the direction of the anode stripes on the MCP is set so as to be parallel to the beam direction, as is shown in the left-hand figure of Figure 4 (Measuring position). For the calibration, however, the direction of the stripes is changed by a pulse motor and set perpendicularly to the beam direction, as shown in the right-hand figure of Figure 4 (Calibrating position). In this position, each anode can be considered to receive the same quantity of ions produced by the circulating beam, and the anode signal should be similar to each other. Therefore, the calibration constants are set by these figure heights.

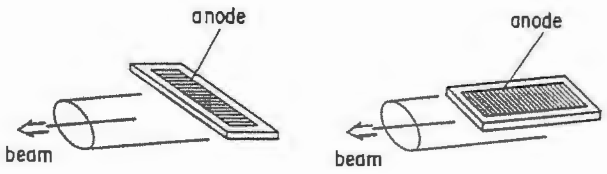


Figure 4. Orientation of the anode strip lines of a MCP.
Left figure: Measuring position (the anode strip are parallel to the beam).
Right figure: Calibrating position (the anode strip are perpendicular to the beam).

D. Rearrangement

The VME computer system takes the time dependence of the current from every anode via an A/D converter in the "measuring position", divides the data by the above-mentioned calibration constants and rearranges them to a "mountain view" of the transversal beam profile within one acceleration period, as shown in Figure 5.

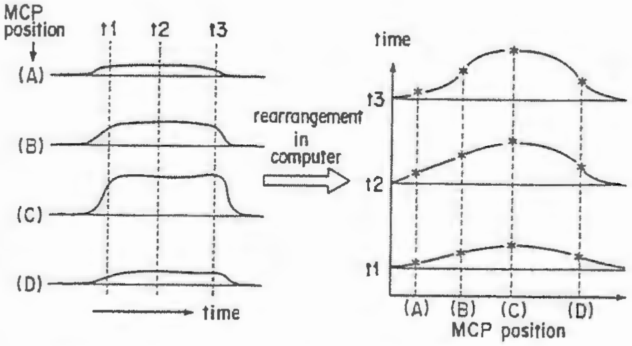


Figure 5. Typical current signal from multi-anodes and a "mountain view" of the beam profile rearranged by a computer.

3. MEASUREMENT RESULT

A. "Mountain view" and time dependence of the beam profile center

The computer outputs by rearranging the data at the place, where the dispersion function is small, of the Main ring. The time dependence of the center of the horizontal beam profile (shown in Figure 7) is in good agreement with the output signal of the ΔR monitor (shown in Figure 8).

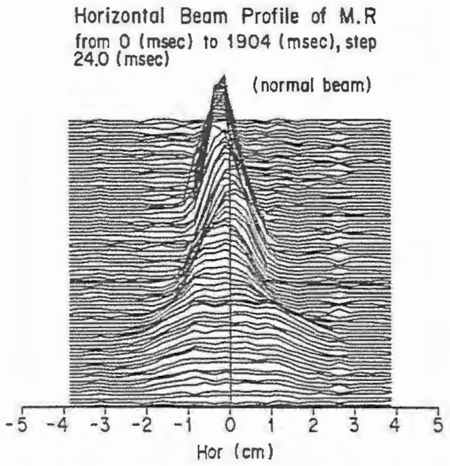


Figure 6. "mountain view" of horizontal beam profile in the Main ring.

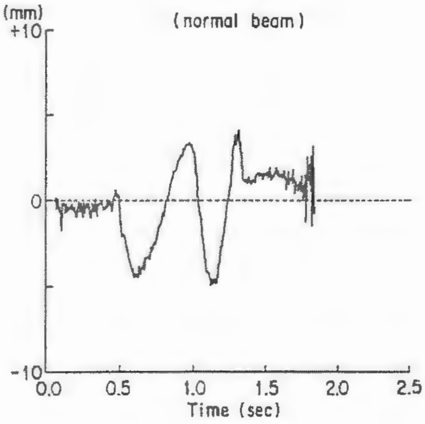


Figure 7. Time dependence of the center of the horizontal beam profile calculated by a computer.

Figure 8. Output Signal of the ΔR monitor at the same position of NDPM (X:200ms/d, Y:2mm/d).

B. Momentum spread of circulating beam

(This measurement idea was suggested by Prof.S.Hiramatsu and Dr.N.Kumagai.)

Assuming that the intrinsic beam profile and momentum distribution have Gausian shapes, the total half beam width (x) is

$$x = (\beta\varepsilon + (\eta\frac{\Delta P}{P})^2)^{1/2}, \qquad (1)$$

where β is the Twiss parameter, ε the beam emittance, η the dispersion function, and ΔP the momentum spread. If two NDPMs are installed at a location with the same Twiss parameters (β), but having a different dispersion function (η_1, η_2), the momentum spread is deduced from the above-mentioned equation to

$$\frac{\Delta P}{P} = (\frac{x_1^2 - x_2^2}{\eta_1^2 - \eta_2^2})^{1/2}, \qquad (2)$$

where x_1 and x_2 are the half beam width at the position with η_1 and η_2, respectively.

The time dependence of a half beam width at 20% height of a beam profile measured by MCP (at small η) is shown in the Figure 9, and the output of EM (at large η) is shown in the Figure 10. The time dependence of the momentum spread in the Main ring is calculated from those two figures and is shown in Figure 11, which shows a sharp peak at the transition time (~0.8s).

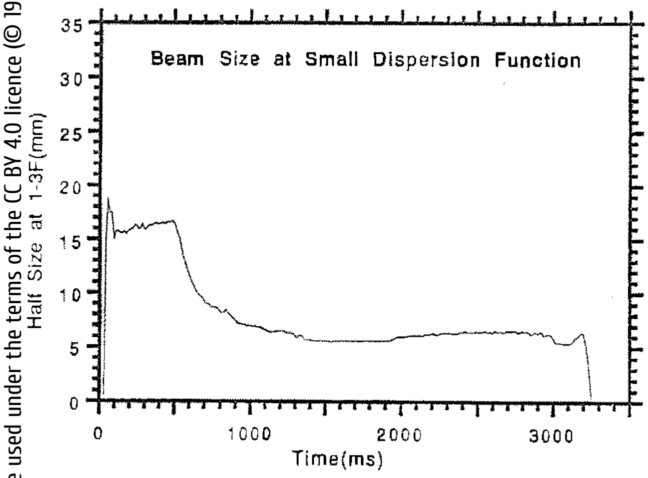


Figure 9. Time dependence of a half beam width at 20% height in the Main ring at small η.

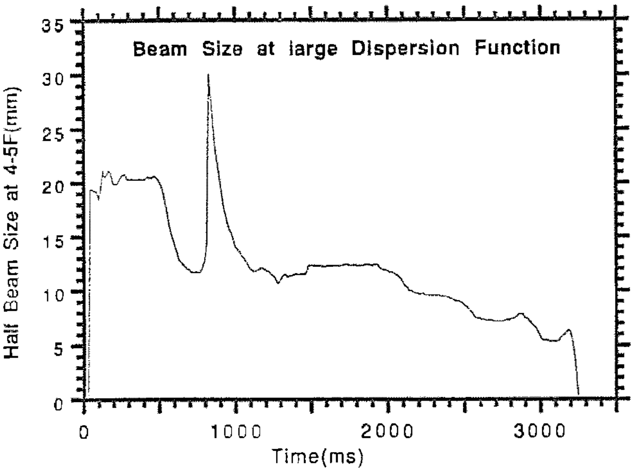


Figure 10. Time dependence of a half beam width at 20% height in the Main ring at large η.

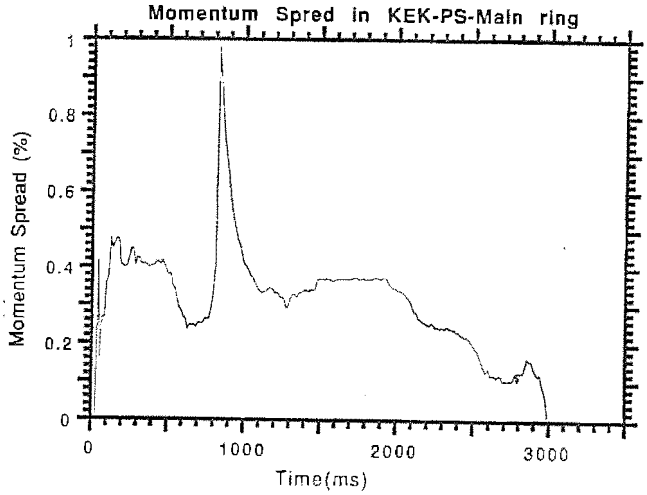


Figure 11. Time dependence of the momentum spread in the Main ring.

4. REFERENCES

[1] T.Kawakubo, T.Ishida, E.Kadokura, Y.Ajima and T.Adachi, "Fast Data Acquisition System of a Non-Destructive Profile Monitor for a Synchrotron Beam by using Micro Channel Plate with Multi-Anodes", NIM A302,pp.397-405,1991.

[2] T.Kawakubo, T.Ishida, E.Kadokura, Y.Ajima and T.Adachi, "Non-Destructive Beam Profile Monitors in the KEK-Proton Synchrotron", KEK Preprint 91-23 May 1991.

A CAMAC-Resident Microprocessor For The Monitoring Of Polarimeter Spin States.

David Reid, Derek DuPlantis, Neil Yoder
Indiana University Cyclotron Facility
2401 Milo B. Sampson Lane, Bloomington, IN 47405

Don Dale
TRIUMF
4004 Wesbrook Mall, Vancouver, B.C. Canada V6T 2A3

Abstract: A CAMAC module for the reporting of polarimeter spin states is being developed using a resident microcontroller. The module will allow experimenters at the Indiana University Cyclotron Facility to monitor spin states and correlate spin information with other experimental data. The use of a microprocessor allows for adaptation of the module as new requirements ensue without change to the printed circuit board layout.

I. Introduction

A custom CAMAC module has been developed to allow for the remote monitoring of polarimeter spin states in the Indiana University Cyclotron Facility . The module will provide experimentalists at the facility with data on spin states of the particle beam and allow them to correlate experimental data with this information. The cyclotron control computer sets the polarized ion source by means of a task which also acts as a network multi-node server, providing spin-state data to client tasks on one or more data-acquisition computers, each of which will copy current spin-state data to the module. The standard data-acquisition programs will access the module while reading other event data.

II. Module Functionality

To meet the experimental demands outlined above the module has two eight bit registers for holding the polarimeter states. One register holds the current state and the second register holds the latched state. Most data-taking events read the current state register. For certain CAMAC commands the latched state is read and the current state is moved into the latched state register.

The experimentalist's interface to the module is through seven output LEMO connectors on the front panel of the module. Three connections give the spin data. One connection gives information on what type of particle (proton/deuteron) is being accelerated through the cyclotron. A ready bit, an interrupt bit, and a valid bit are also brought out to the front panel.

A sixteen bit timer countdown register is used in the module. This register counts down in 0.1 second units. This register is loaded by a CAMAC write operation and begins counting down immediately upon being loaded. When the countdown register makes the 1 -> 0 transition the module, current state register, and the valid outputs are cleared.

A sixteen bit sequence number register holds polarization cycle sequence number to permit correlation of event streams from different data acquisition computers. The sequence number register is specified by the data acquisition computer and written to the module.

III. Module Components

i. Microcontroller

To meet the functionality requirements for the module it was decided to use a resident microcontroller. The use of the microcontroller will allow the module to meet the current specifications and give additional flexibility to meet future demands. By using a microcontroller the module can be readily adapted to future uses without the costly printed circuit board redesign that would result from the use of logic circuitry.

The microcontroller that was chosen for this project is the Intel 80C196KC. The chip has a sixteen bit wide internal data bus. Because the registers for the module are specified to be eight or sixteen bits wide the internal data bus for the chip allows for direct, full width register operations.[1]

The module can perform complete operations before the next CAMAC cycle due to a 16 MHz clock.[2]

ii. External Memory

The 196 can take advantage of external memory devices. This features makes the use of external ROM and RAM onboard the module possible.

The module uses three external memory devices. Two 8K x 8 EPROMs are used to hold the code for the microcontroller. The microcontroller accesses the code for its internal operations from the EPROMs. Two chips are used in parallel to allow for sixteen bit wide memory words to be used.

Two 8K x 8 RAM chips are used for external memory register space for the microcontroller. For the functionality of the module as now specified this additional

RAM space is not necessary. However, the decision to incorporate this memory space was made to provide for unforeseen modifications to the requirements for the module. By designing in this extra memory the module will be able to do more complex data and register operations in the future. Again, because of the RAM chips memory structure, two chips were used to allow for sixteen bit data words.

One Integrated Device Technology 7133 Dual Port RAM is used as register space for data that will be written onto the CAMAC data bus. The use of this is necessary to meet the CAMAC specifications for dataway operations. The microcontroller (even with a clock frequency of 16 MHz) cannot respond to a CAMAC write command fast enough to have the appropriate data present on the CAMAC dataway when the S2 line is asserted 700 ns into the CAMAC data cycle.[3] The IDT chip was chosen because of its 55 ns address access time.[4] This speed will allow for valid data to be present at the dataway for the read command.The IDT chip has a sixteen bit wide memory structure.

The usage of a dual port RAM chip is acceptable because of the functionality requirements of the module. No data manipulations must be done on the data present in the registers in the 700 ns from the beginning of a CAMAC data cycle to when valid data must be present on the dataway.

To load the appropriate register data onto the dataway within 700 ns an EPLD is used to decode the CAMAC Function and Address lines. This EPLD logically recognizes the CAMAC read commands and then sets the address to point to the correct memory location in the Dual Port Ram. When the S2 line is asserted the Dual Port then writes the requested register data onto the CAMAC dataway.

The EPLD also sets the specified values for the Q, X, and Look-At-Me (LAM) lines. Use of the EPLD insures that these signals are present on the dataway at the appropriate times in the CAMAC cycle. The Q line can be asserted or deasserted, or can be set to reflect the state of the ready bit or the LAM. The EPLD also sets X=1 for all valid functions.

iii. Data Latches

Because of the timing constraints of the CAMAC dataway, cycle valid data from a CAMAC write operation will not be present on the dataway by the time the microcontroller can respond.[5] To solve this problem, 74AS573 Transparent Latches are used to latch in the write data and hold it until the microcontroller is ready to process the information. The data is latched off the bus by the logical AND of the N and S1 lines. This insures that valid write data is held for the microcontroller.

The microcontroller memory maps the write data latches into its external memory structure. When the microcontroller is ready to process the write data, it asserts the correct address lines and processes the information.

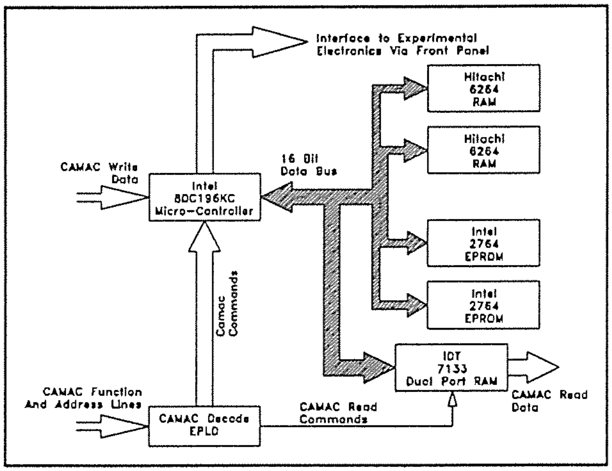


Figure 1 Module Block Diagram

These latches also store the CAMAC Function and Address lines. This information is held until the microcontroller can process and decode the commands.

All external memory decoding will be handled by a second EPLD. This EPLD decodes the address bus from the microcontroller and asserts chip enable lines to each external memory device. Addresses were specified to allow for use of the entire memory space of each external device in the future.

IV. Camac Functions

The module has been designed to respond to the following commands, while setting Q and LAM lines accordingly.

F(0)A(0)	Read entire Polarity Register, set Q to the value of the ready output pin.
F(2)A(0)	Read entire Polarity Register, latch the current state into the latched register, set the ready output pin to the value of the 5th bit of the latched register, clear the LAM and interrupt pin, and set Q to previous ready output pin.
F(0)A(4)	Read Timer register, assert Q line.
F(0)A(6)	Read Sequence register, assert Q line.
F(9)A(1)	Clear the current Polarity register, clear the timer register, clear ready bit, set LAM and interrupt output pin, set Q to LAM value.
F(16)A(1)	Write current state, clear the ready bit, assert LAM, set interrupt bit, assert Q.
F(16)A(4)	Write timer register, assert Q.
F(16)A(6)	Write sequence register, assert Q.
F(24)A(0)	Disable LAM, deassert Q.
F(26)A(0)	Enable LAM, deassert Q.
F(8)A(0)	Test LAM, set Q=LAM.
F(10)A(0)	Clear all registers, clear LAM, set Q=LAM, clear interrupt bit.

F(16)A(0)	Write entire polarity register, assert Q.
F(27)A(1)	Test interrupt bit, Q=interrupt output.

The module will enter appropriate reset states by the assertion of the Z, I, or C lines. The Z signal resets the module by clearing all registers, disabling the LAM, and stopping the countdown timer. The C line clears the registers and stops the timer. The I line inhibits the module and precludes any processing while the line is asserted.

V. Software

Programming, written in Assembly, for the microcontroller is stored in the EPROM chip. Software development for the module will be done using the Intel EV80C196KC Evaluation Board and 2500AD 8096 Assembler software on a PC. The EPROM chips will then be burned with the code for the module.

The microcontroller is in an idle state until the module is addressed. An interrupt to the microcontroller is generated by the N line of the CAMAC dataway. This interrupt causes the microcontroller to access the EPROM memory vector and begin executing the program. The microcontroller will then access the F()A() latches. The controller decodes these and proceeds with the proper operations. During all program executions the microcontroller uses a high speed output pin to assert the X line as a module busy line. This alerts the computer is still processing and avoids the possibility of the module receiving a second interrupt before the processing of the first command is complete.

VI. Status

As of this writing the module is ready to enter the testing phase. The design of the circuitry and printed circuit board layout has been completed and the module has been assembled. Software development for the microcontroller needs to be completed and extensive testing is required before the module is ready to be incorporated into experimental setups.

VII. Acknowledgements

The authors wish to acknowledge the contributions to this work by T. Capshew, J. Collins, D. W. Frame, J. Graham, and the Computer/Electronics group at Indiana University Cyclotron Facility.

This work was done under the support of National Science Foundation Grant #NSF PHY 90-15957.

VIII. References

[1] Intel, *16-Bit Embedded Controllers Handbook*, Mt. Prospect, IL, 1990, pg. 5-1 - pg. 5-126
[2] Peter Clout, *A Camac Primer*, LA-UR-82-2718, Los Alamos National Laboratory, 1982, pg. 72 - pg. 74
[3] IEEE Std 583-1975 pg. 50
[4] Integrated Device Technology, *High-Speed CMOS Databook*, Santa Clara, CA, 1988, pg. 5-61
[5] Peter Clout, *A Camac Primer*, LA-UR-82-2718, Los Alamos National Laboratory, 1982, pg. 33

High accuracy measurement of magnetic field in pulse magnetic elements

V.Kargaltsev, E.Kuper
Institute for Nuclear Physics,
Novosibirsk, 630090, USSR

Abstract

CAMAC module intended for measurements of instant magnetic field using coil sensor is described. It is four channel integrating ADC with current input in which signal integration time is controlled externally and may be optimized for a given signal. Original technical solution allowing to eliminate influence of the integrator capacity and switches instability on overall accuracy is described.

The large accelerator facilities include a great number of magnetic elements interacting with a beam for a short period ranging from 0.01 ms to 10 ms. For example, this class of elements includes all the magnetic components of channels for particle transportation. In addition, most of these elements are operating rarely - once in 1 - 10000 s. For these elements the most optimal is the use of a pulse power supply that reduces the electric power consumption and which is most important, it solves the problem of heat removal. Though, the pulse power supply poses some problems in providing the accuracy of magnetic field and its measurements.

In practice, the measurement problem can be reduced to the measurement problem of instantaneous value of the magnetic field. In fact, the time of the beam-field interaction is usually so short then the field can be taken quasistatic and acting equally on all the portions of a bunch of particles.

There are some elements interacting with a beam for a long time during which the field can be changed substantially. For example, the cyclic accelerators operate in the similar way. But the pulse shape in these elements is determined by the properties of the feeding generator and it is very conservative. The shape relevance can be checked by the point by point measurements while development of such an element and during the operation it is sufficient to control one or two characteristic points (instantaneous value) of a pulse. Usually, the values are measured which correspond to the beginning and the end of the field interaction with a beam.

The inductance sensor proved to be very convenient for the pulse measurements. It can easily allow the shielding and galvanic de-coupling from the facility construction that facilitates substantially the problem of producing the measuring devices.

The experience of operation of the facilities at the Novosibirsk Institute for Nuclear Physics (INP) has shown that at the requirements to the accuracy of magnetic field lower than 0.05% the tuning of magnetooptic channel was determined by the measurements of fields with these sensors. At higher accuracies one should take into account the deviations between the field (flux) value measured with the help of this probe and the properties of magnetic element as a whole, which are caused by the magnetic temperature variations and some other reasons.

While measuring the instantaneous value the following approach seems to be natural: the field signal is traced with the analog memory device, stored in the memory at the moment of interest and then it is transformed into the code.

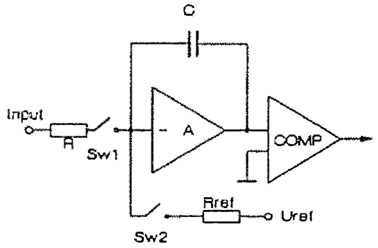


Fig.1. The analog section of module.

The use of the inductance probes enables one quite simple to realize this process with the help of the analog integrator. In fact, the signal voltage from the inductance probe is proportional to the speed variation of the passing magnetic flux:

$$E = W \frac{d\Phi}{dt}$$

Here W is the number of turns of a probe.

If this signal is integrated by the analog integrator, the charge stored in the capacity can be described as follows:

$$q=\int_{t_0}^{t_x} \frac{dE}{R} dt = \frac{W}{R} \int_{t_0}^{t_x} \frac{d\Phi}{dt} dt = \frac{W}{R} (\Phi(t_x)-\Phi(t_0))$$

In this case, the integration limits are given by the moments of connection (t_0) and disconnection (t_x) of the switch Sw1. If the integration is started before the field pulse, the stored charge is equal to:

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S11LLC10

$$q(t_x) = \frac{WS}{R} B(t_x)$$

B is the magnetic inductance
S is the sensor cross-section

The product W * S characterizes the sensor sensitivity.

For converting the charge into the code the capacity C is discharged by the calibrated current on the circuit Uref, Rref and Sw2 and the discharge time indicated by the comparator is then measured. As a result, the value

$$N = \frac{q(t_x)}{Iref} = \frac{W\ S\ Rref}{R\ Uref} B(t_x)$$

is uniquely related to the magnetic field value at the moment when the integration is stopped. From it is seen that the accuracy of measurements is determined by the stability of the sensor parameters (W * S) and three elements of the measurer: R, Rref, and Uref. In order to provide the high accuracy the wire and the metal-film resistors of highest stability are used as R and Rref. The voltage reference diodes have passed the preliminary test, certified of the thermostable point. This mode enables one to avoid the use of the oven for the diode at the accuracy of the reference voltage of up to 0.002%. As to the capacity C, its nominal is not important. The only thing required is its short-term stability. The leakage currents of modern capacitors are negligible small and the only error, which could be introduced by the capacitor is the variation of the effective charge caused by the adsorption (the polarization of the dielectric of capacitor). In order to achieve the accuracy about 0.01% it is sufficient to use the capacitors with the low polarization dielectrics (polystirol, teflon). In order to reach the higher accuracies one has to use the special analog circuit of the compensation for polarization. The parameters of the circuit are selected for each certain capacitor.

As a rule, the problem of precise measuring the time interval does not make any difficulties.

Some difficulties occur with the switch Sw1. To provide the time for commutation (about 10 ns) requires the use of semiconductor switches which have the noticeable switch-on-resistance. The switch resistance is added to the integrating resistance R and its instability introduces the error. This problem is solved by the "triangle" of switches Sw1 (Fig.2) assembled with the field transistors.

In this scheme, the on-state of the switch S1a, S1c and off-state of S1b correspond to the on-state of Sw1. While the integration of a signal the current passes through the switch S1a but the amplifier watches the point G via the switch S1c and namely at this point provides the zero potential. Thus, the voltage drop on the switch (in this case, on S1a) does not make any influence on the charge stored in the integrator capacity. During the storage and conversion the switch Sw1 is disconnected (S1a and S1c are off, S1b is in the conducting state providing the connection of the feed back circuit).

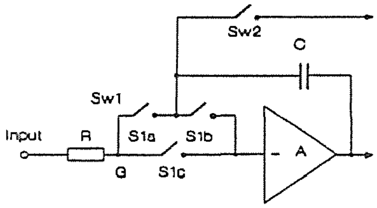


Fig.2. The integrator with "triangle" of switches.

The typical current of a signal is of 0.1-10 mA. Therefore, the operational amplifier with the field transistors with the input current lower than 0.1 nA with an extra circuit of dynamical correction of the input bias voltage fits quite well as the integrator amplifier.

For carrying out the practical measurements, requiring, as a rule, the simultaneous measurements of a few tens of signals the four-channel CAMAC-module is designed that is operated according the principal described.

The field sensors are located just in the magnets and they are connected to the measuring devices with a coaxial cable. The connecting cable can be up to 100 m long with substantial capacitance. To avoid the distortion of the signal by the capacity of the cable and thereby to avoid errors, the integrating resistor is removed to the sensor. In this case, the measurer has the zero input resistance, which cancels the influence of the track capacity.

The moments of the integration start and stop are given from outside and usually by the pulses of synchronization system of the facility under service. In this case, the input "integration start" is common for four channels and the control for the integration stop is individual for each channel.

Technical parameters of the module:

Number of channels	4
LSB weight	5 pQ
Dynamic range	16 bit
Accuracy	0.01 %
Conversion time	50 ms

The given accuracy is realized at the integration time of a signal of longer than 500 μs, at lowest times the accuracy decreases.

At present, the apparatus is modified in order to improve the measurement accuracy (an accuracy nearly 0.002%, scale 18 bit).

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S12FC01

FEEDBACK -- CLOSING THE LOOP DIGITALLY

J. Zagel, B. Chase
Fermi National Accelerator Laboratory*
P.O. Box 500
Batavia, IL 60510

Abstract

Many feedback and feedforward systems are now using microprocessors within the loop. We describe the wide range of possibilities and problems that arise. We also propose some ideas for analysis and testing, including examples of motion control in the Flying Wire systems in Main Ring and Tevatron and Low Level RF control now being built for the Fermilab Linac upgrade.

I. INTRODUCTION

The standard techniques used to design and analyze analog feedback systems can also be applied to digital systems. It is desirable to consider frequency response, maximum tolerable error, and stability questions for systems controlled by processors. In modern digital systems a considerable amount of software not only replaces analog circuit functions but also allows additional features to be built into the system.

II DEFINITIONS

A. Control System

A control system is generally described as a system that provides an control output variable C in response to an input reference R. This can be accomplished open loop or closed loop. Open loop control means that for a given input, the plant G provides a fixed response regardless of any external loading on the controlled device or process. Closed loop control uses feedback signal H to compare the output to the reference input and generate an error E which is then minimized by the loop.

A predictor of the desired output can be applied to the drive circuit thus producing a feed forward signal. Predictions are normally obtained by computations on a mathematical model combined with measurements of the actual process.

The plant G can be viewed as the combination of fixed drive characteristics plus an equalization, or compensation, filter applied to correct any undesirable behavior. The feedback can be a simple transfer function, such as position to voltage, or a complex filter to aid in measuring the controlled process.

An open loop system is described as the convolution of r(t) with g(t) in the time domain. It is more convenient to analyze these systems in the frequency domain using Laplace transforms. This transforms convolution integrals to multiplication for continuous, linear systems, or $C(s) = R(s) * G(s)$. With feedback, the transfer function is described for the closed loop configuration to evaluate the response and stability of the system. For a closed loop: $\frac{C(s)}{R(s)} = \frac{G(s)}{(1+G(s)H(s))}$

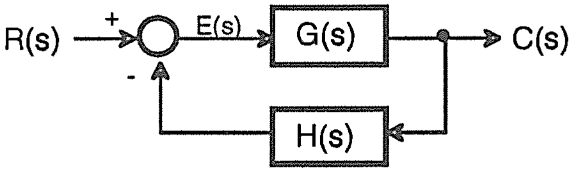


Figure 1. Basic Control System Terminology

B. PID Loops

The most common controller is the proportional - integral - derivative loop (PID). To understand the PID loop we will look at the pieces of a motion control system. When a position change is required the reference input to the system is modified. The system will generate a drive signal proportional to the position error developed between the now changed reference input and the previously held position.

For a step change in the reference input, the drive electronics may allow the motor to far overshoot the desired change. In this case it is useful to consider the first derivative term of the velocity, or acceleration, to maintain stability. This is also referred to as lead compensation.

To correct for long term or steady state errors in the desired output a third, integral, term is included in control loop. This term removes accumulated error over time. This is referred to as lag compensation.

The mathematical formulation[1] for the PID filter in time is: $u_{PID}(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de}{dt}$

which transforms to:

$$U_{PID}(s) = K_p + K_i \frac{1}{s} + K_d s$$

$$= \frac{K_d s^2 + K_p s + K_i}{s}.$$

This filter function has one pole at the origin and two zeros that are dependent on the three gain terms.

C. Hardware - Software Equivalents

To implement the PID equation above active elements are used. The hardware is shown in figure 2. Each term is shown as an independent active element however in practice this circuit can be simplified.

While Laplace transforms take us into a convenient domain to analyze analog circuit behavior, the z-transform better serves the transition into sampled time domain.

---

*Operated by Universities Research Association Inc., under contract with the U.S. Department of Energy.

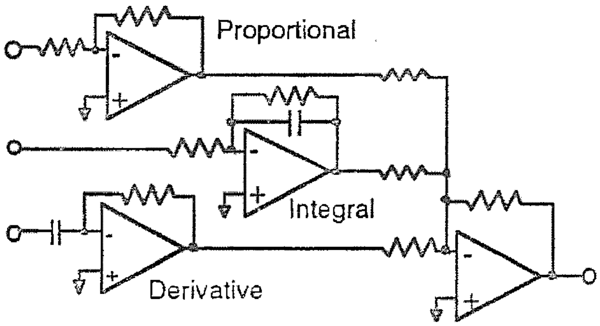


Figure 2. Analog Hardware Elements of PID Filter.

The transform: $s = \frac{2(z-1)}{T(z+1)}$ and $z = \frac{1+\frac{sT}{2}}{1-\frac{sT}{2}}$ known as the bilinear transformation, applied to the Laplace yields the z domain.

The function expressed in the z domain assumes a sampling in time where the sample interval is constant. The filter function is then transformed into a series of outputs correlated to the n^{th}, n - 1, n - 2, etc. samples in time. This yields a difference equation for the filter output :

$$u(t) = u(n-2) + K_1e(n) + K_2e(n-1) + K_3e(n-2)$$

where $K_1 = K_p + 2K_d/T + K_iT/2$,

$K_2 = K_iT - 4K_p/T$,

and $K_3 = 2K_d/T - K_p + K_iT/2$.

These constants can then be programmed into a digital signal processing system. [2]

D. Stability

The typical stability criterion applied to analog systems are Bode plots in frequency, and s-plane ($s = \sigma + j\omega$) plots for Laplace transfer functions. Bode plots graph the gain and phase of the system vs frequency to determine both gain and phase margin. A stable system has a gain less than 1 when the phase reaches 180^o.

The s-plane plots the imaginary and real parts of the Laplace transfer function. The requirement for stability is that all roots of the characteristic equation have negative real parts. Using the z-transform for discrete time sampling systems the z-plane stable area maps into the unit circle.

III. MOTION CONTROL

A. Introduction

Typically motion control systems fall into two categories. A velocity control system tries to maintain a continuous speed profile for a system while a position control system will try to first establish and then maintain a position corresponding to the desired input. The Fermilab flying wire system described previously[3] uses a combination of these to maintain position and control the velocity profile of the system when a position change is required.

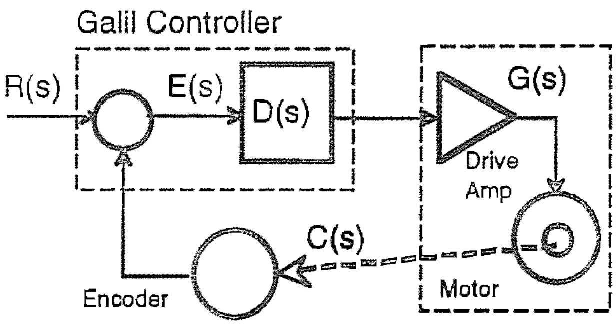


Figure 3. Flying Wire Control Loop

B. Velocity Profile

The position change of the wire is accomplished by accelerating at a constant rate, holding a constant velocity, and then decelerating at the same constant rate used for acceleration. The Galil Controller contains a digital filter[4] with $D(z) = GN\frac{z - \frac{zr}{256}}{z - \frac{pl}{256}}$ which transforms to $G(s) = K\frac{s+a}{s+b}$ which, for a<b, is a lead filter. K = GN and the zero and pole term relate to a and b as $GN\frac{256 - zr}{256 - pl} = K\frac{a}{b}$.

The wire speed in the Fermilab Tevatron is $5\frac{m}{s}$. The wire is mounted on a fork that is .0965 m radius yielding an operating frequency of 325 radians/sec. The optical encoder provides 16384 counts/rev. Hardware prescales the counts of the encoder and the processor periodically samples the output to determine the actual position error on a time scale that matches the required system performance. Typically a sampled system should run at 10 to 20 times the minimum sample rate for the measurement process. It is clear that the processor can not directly sample the encoder inputs in real time. However the motion of the wire can be corrected on the millisecond timescale required of the system.

C. Tuning the Loop

We want to select the parameters to produce a fast but stable response. For the flying wire system the most stable set of operating parameters have historically been determined empirically. We have since taken measurements with an HP3563A Control System Analyzer to determine the true best parameters over a wide range of operating conditions.

From these measurements the natural oscillation frequency can be obtained. The zero is then selected to be half to two thirds of the natural frequency such that: $ZR = 256e^{-0.4\omega_cT}$ where T is the sample interval. For an ω_c = 200 rad/sec and T = 500 μsec yields ZR = 246. Our zeros are in the range of

245 to 248. The digital number input is 0 to 255. The gain varies from system to system but in the range of a few counts (min 4 to max 7) out of 256.

The mechanical components of the system are subject to change due to temperature, humidity, and wear. If the closed loop parameters are set appropriately then the system will operate reliably over time. For our system the pole is set at its default value.

D. Control System Analyzers

There is a new class of instrumentation available to aid in the process of designing and implementing closed loop systems referred to as Dynamic System Analyzers. These analyzers work in both time and frequency domain. A special case is the Control System Analyzer (CSA) which also allows inputs or outputs to be either digital or analog. Once the measurement has been made, a wide range of math functions is available for analysis.

The CSA has been used to make step response measurements in the time domain. It will be used to make measurements of the closed loop performance of the system. This work has not yet been accomplished.

IV. LINAC LLRF

A. Introduction

The last three 201 MHz accelerator sections of the Fermilab linear accelerator are being replaced by seven new sections operating at 805 MHz, each driven by a 12 MW Klystron. Due to the higher frequency and higher gradient this upgrade will increase the beam energy of the linac from 200 to 400 MeV. In order to minimize momentum spread of the beam, tight regulation of the RF gradient in the cavities is needed. Feedback alone does not have the needed bandwidth to meet the regulation specification, therefore a processor based learning feedforward system was added.

B. The System

The accelerating field gradient in the cavity is regulated by a feedback system controlling the magnitude and phase of the RF field inside the cavity. The design specification is to regulate the gradient field to 1% magnitude and 1 degree phase during the time that beam is present. This requirement for regulation is made difficult by three aspects of the RF system. First, there are long group delays, (400ns), through the Klystron and wave-guides connected to the cavity. The Laplace transform of this delay(Td) is e^{-sTd}. This is a frequency dependent phase shift that limits the closed loop bandwidth of the feedback loop to less than 400KHz. Second, the beam loading is a rectangular pulse that has a very fast rise time. This demands a fast response from the control loops. The third problem is that while the cavity responds like a simple LC resonator at its central resonant frequency, it can also be excited in other modes at frequencies that are within ±500KHz. Any positive loop gain at these resonant frequencies will make the system unstable.

A fast feedforward loop that is able to learn from the past history of accelerating cycles is needed. We chose a digital system to generate the feedforward waveform because of its versatility and accuracy. Digital signal processing techniques enable the use of a learning algorithm. Because true system testing will not be done until all installation is done and there is beam in the machine, the hardware needs to be made as general as possible. Any last minute modifications can then be done in software. A block diagram of the RF system is shown in figure 4. Note that all the electronics for the low level system resides in the VXI crate.

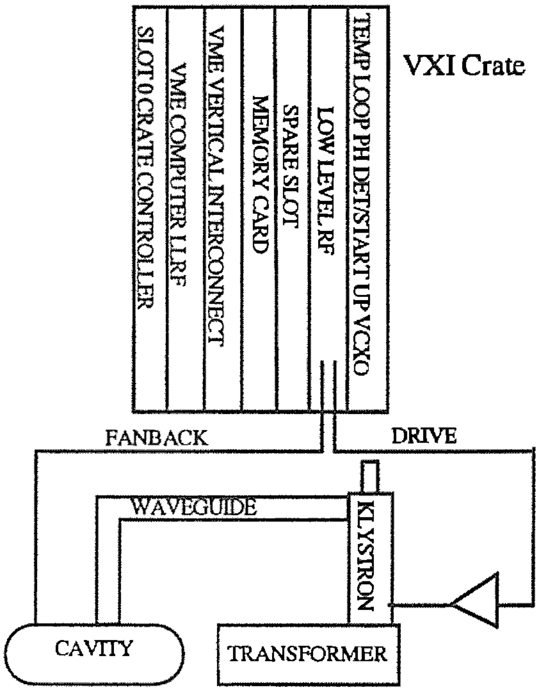


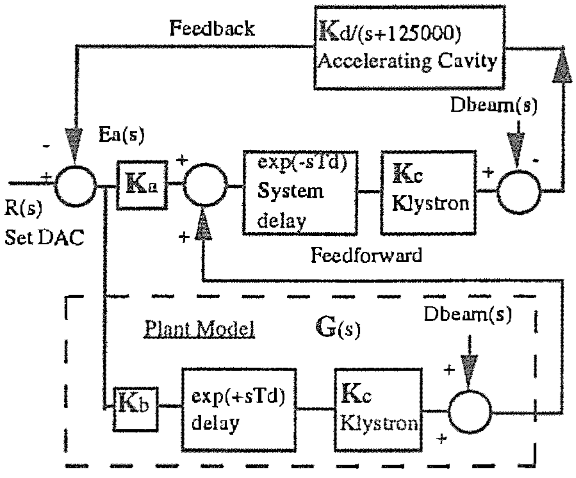
Figure 4. LINAC RF System

C. The Model

The system was modeled to a great degree of accuracy using a time domain analysis program, EXTEND[5]. Using EXTEND, the nonlinear components such as the Klystron, mixers and phase shifter are modeled so that the system can be understood over its full dynamic range of operation However a much simpler linear model works well over a narrow dynamic range. This linear model is shown in Figure 5.

All signals shown in the diagram are modulation signals of the 805 MHz signals magnitude or phase. The 805 MHz carrier is mixed in and out with the control signals, but has no direct relation to the feedback loops.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S12FC01



Ka is feedback loop gain.
Kb is a "learned" gain by an IIR filter.
G(s) is a heuristically determined function.
Dbeam(s) is the disturbance due to the beam
Feedforward is active only during beam.

Figure 5 Linear Model of RF System.

The feedforward system was dictated by the conflict between the need for fast system response (200 ns) and large group delay (400 ns) due to the Klystron and waveguide. Feedforward is made possible because the system, running at 15Hz, is very repetitive from one acceleration cycle to the next. This enables the algorithm to learn the gain K_b by the use of an IIR Filter that is incremented once each cycle. Therefore, the filtering function is in the domain of cycle to cycle time and not the linear time of the 125us of one accelerating cycle.

G(s) is the plant model for the transfer function of the beam loading effect and Klystron combined. G(s) can be any function or array that is found to be the best fit and is limited to only a 2.5 MHz bandwidth by an anti-aliasing filter.

D. The Hardware

In order to implement the feedforward system, a fast waveform capture and playback generator is needed. The amount of signal processing and I.O. needed demanded a high-speed dedicated processor and wide-band computer bus. Good RF shielding at 805Mhz and a quiet environment for signal conditioning is necessary. The system also needs to be remotely operated and must have good remote diagnostics. Due to the large number of components a large card form factor was needed. VXIbus was chosen as the platform, as it was conceived for this general type of application. Figure 6 is a block diagram of the Low Level Module.

The control system is designed to be as flexible as possible, since full testing of the system will not be done until there is beam in the machine. This philosophy is embodied by giving the processor full access to the wideband error signals of the feedback loops and 16 circuit diagnostic points that may be sampled at any time during the 125 us of the RF pulse.

The error signals for both the magnitude and phase loops are sampled at 10 Mega-samples/s by an 8 bit ADC and stored in FIFO memory. After the error waveforms have been digitized, the module generates an interrupt that starts the DSP routines in the processor. The computed feedforward waveform is then loaded into another FIFO memory and played back during the next accelerating cycle through a 12 bit DAC. This signal is summed with the feedback signal before driving the phase shifter or the magnitude modulation mixer. The 8-bit data from the ADC is averaged by the CPU which gives a higher resolution to the playback DAC.

E. The Operating System

The MTOS-UX kernel from Industrial Programming, Inc. is a true multi-processor multi-tasking real-time operating system. This kernel is in wide use at Fermilab and has been ported to many different platforms.

The system is designed so that on startup, the software detects the number of low level RF modules and processors that are installed in the VXI mainframe. It then configures itself to distribute the processing capability as needed by each LLRF system. This allows the system to be reconfigured by users without the support of a system software expert.

F. The Software

The main job of MTOS (see Figure 6), is to complete two main tasks when it receives an interrupt from the LLRF module. The interrupt starts the producer task "I_15HZ", which then reads the 16 MADC channels and the two ADC FIFO memories. Next, it reads the control buffer memory and then writes to the LLRF Module registers. Finally, it places pointers to the data arrays in the MTOS message buffer, and then unblocks the consumer task M_15HZ.

"M_15HZ" processes the error waveform, creates the feedforward waveform, and writes it to the DAC memory in the LLRF module . It then resumes waiting on the message buffer for the next task. These tasks must be completed before the next interrupt is received.

The task "TEST" keeps track of the percent of idle time for the processor and displays it on an SSM module. This allows the developers to monitor processor bandwidth as changes are made. This has proved to be a very useful function.

"DIAGNOSTICS" is off-line code that will test many of the signal paths in the module. This includes memory, ADCs, DACs, MADCs and some of the RF paths. The human interface for this will be LabView running on a Macintosh 2ci. Communication from the Macintosh to the LINAC control system is done over TokenRing.

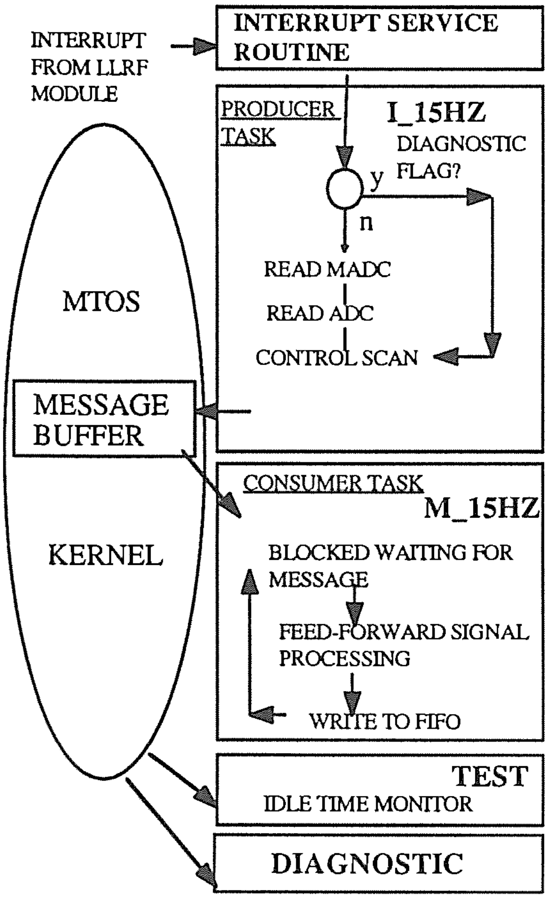


Figure 6 LLRF software overview

G. Feedforward

Originally the concept was to have the feedforward loop learn the entire error waveform. Each sample would have an IIR filter that would learn the best value for that point in time to minimize the error signal. In the model this worked well as all the system delay could effectively be removed from the loop. This would allow the effective system closed loop bandwidth to be increased to about 2.5MHz, the frequency of the anti-aliasing filters. This algorithm continuously adapts to any changes of the system. The main problem with this approach is the other resonant modes of the cavity that are within ±2.5 MHz of the main resonant mode. These modes are ideally nulled out of the fan-back signal by summing all 8 cells of the cavity together. However, mismatches in the signal combiner cause gain slope changes in the response that cause the closed loop system to oscillate. Another approach is needed in order to achieve the needed closed loop response.

The beam profile is very close to a rectangular pulse as it is run through a beam chopper after the ion source. See figure 7. To model the beam loading effect, the only information needed is the start time, stop time, and the amplitude. If the error waveform does not change except in amplitude, a simple software routine can create the proper shape of the feedforward signal. The amplitude can be "learned" by looking at past error waveforms. Thus fast changes can be made to the drive signal, (100ns), without affecting the closed loop bandwidth.

The proper amplitude of the feedforward pulse is computed in the following manner. A section of the error waveform, before the beam, is averaged in order to establish a baseline value. In the same way the average error is found while beam is present. The beam present value is subtracted from the pre-beam value and multiplied by a gain constant. This value is then filtered over many accelerating cycles by an IIR filter. The filter algorithm is simply 5% new value plus 95% old value. This filtered value is then added to a baseline value of the feedback waveform between the specified start and stop times of the beam. When the loop comes into its steady state value the error waveform will be flat across the top. This is because the feedforward loop is doing all of the beam loading correction while the feedback loop is taking care of slower errors.

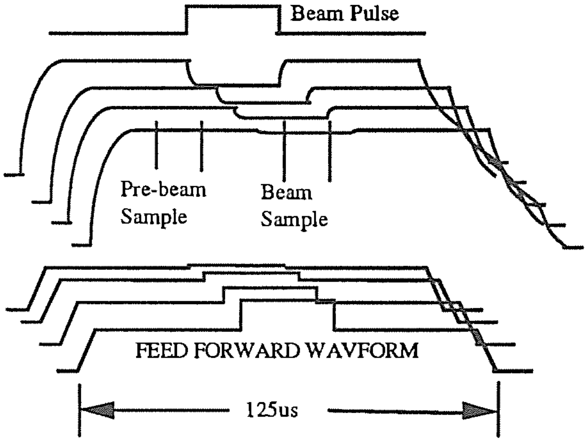


Figure 7 Feedforward learning over several pulses.

In this system, the processor is not in the feedback loop, therefore processing time does not create phase shifts and possible loop instabilities. Feedforward, in the time frame of a single pulse is an open loop process. Therefore loop stability is not an issue as long as the processor gets the job done in the 66ms before the next cycle.

H. Results

The Low Level RF system has been operated on a complete system test bed in the Fermilab A0 lab. It responds very closely to the model with few surprises. One area that is not accurate in the model is the complex interaction of the Klystron, waveguide and cavity. True beam loading effects have not been seen as of yet, however, system settling time to 1% was measured to be less than 300 ns for a 20% step change introduced by a test system. The versatility of the system was demonstrated when system parameters for tuning were changed on the fly from the parameter page. Problems such as the saturation of ADC's or DAC's were easy to find by using the quick plot routines. The LLRF system was operational at A0 in the first week after installation.

V. CONCLUSION

The use of feedforward for the LLRF system allows flexibility in the design and implementation. Once the system becomes operational some minor changes in the feedforward parameters or the algorithm can easily be made.

For the Flying Wire system we hope to generate an optimal set of parameters with a single command to the system. The processor should be able to manipulate the motion control system to determine the actual friction, inertia, and loading of the entire system as installed . When a failure occurs that requires component replacement, the system will be programmed to self tune the loop for the best response.

VI. REFERENCES

---

[1] DiStefano, J. et. al. *Feedback and Control Systems*, McGraw-Hill, 1990, or other Control Systems Texts.
[2] Ahmed, I., "Designing Control Systems", *Digital Signal Control Applications with the TMS320*, Texas Instruments, 1991, pp 48-52.
[3] Gannon, J.,et al., "Flying Wires at Fermilab," Proceedings of the 1989 IEEE Particle Accelerator Conference, pp. 68-70, March 20-23, 1989.
[4] Tal, J., *Motion Control Applications*, Galil Motion Control Systems, 1989.
[5] EXTEND copyright by Imagine That, Inc. San Jose, CA, USA. Details of analysis in Pasquinnelli, R., "Modeling of the Linac Upgrade RF System", Fermilab internal memo, February 6, 1991.

Generalized Fast Feedback System in the SLC*

L. Hendrickson, S. Allison, T. Gromme, T. Himel, K. Krauter,
F. Rouse,† R. Sass and H. Shoaee

Stanford Linear Accelerator Center, Stanford University, Stanford, CA 94309

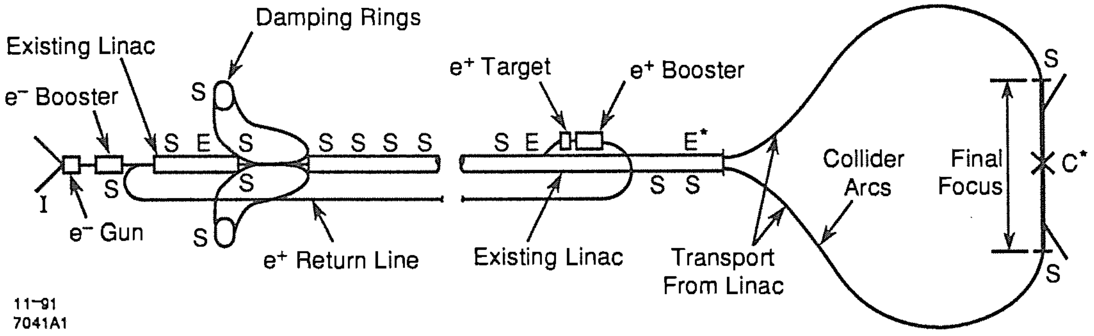


Figure 1: Layout of the SLC with fast feedback loops shown. **S** = steering loop; **E** = energy control; **I** = intensity control; **C** = special-purpose loop to maintain beam collisions; * = prototype.

Abstract

A generalized fast feedback system has been developed to stabilize beams at various locations in the SLC. The system is designed to perform measurements and change actuator settings to control beam states such as position, angle and energy on a pulse to pulse basis. The software design is based on the state space formalism of digital control theory. The system is database-driven, facilitating the addition of new loops without requiring additional software. A communications system, KISNet, provides fast communications links between microprocessors for feedback loops which involve multiple micros. Feedback loops have been installed in seventeen locations throughout the SLC and have proven to be invaluable in stabilizing the machine.

INTRODUCTION

The SLAC Linear Collider (SLC) produces pulsed bunches of electrons and positrons which are accelerated in a LINAC and steered around arcs before colliding at a single interaction point. The maximum beam rate for the machine is 120 Hertz. The SLC control system is based upon a central DEC VAX 8800 and a series of Intel 80386 microprocessors (micros). The micros are distributed geographically, with each micro controlling the devices which accelerate, steer and measure the beam in a region of the machine. The VAX communicates with the micros through a specialized network system, SLCNET, but with the exception of this fast feedback system the micros do not ordinarily communicate with each other.

The feedback system is used for controlling the energy, trajectory and intensity of the beams. The system takes measurements, calculates state functions and implements corrections at a fast rate. It is designed to operate at the beam rate but due to CPU limitations it operates at a lower rate, typically 20 Hertz. Figure 1 shows the SLC machine with currently implemented and planned feedback loops. Prototype feedback systems were initially implemented in three locations for steering, controlling the beam energy [1] and maintaining collisions [2]. These systems quickly became indispensable to the machine operation and an improved, database-driven system was developed to allow easy addition of new loops throughout the machine.

---

*Work supported by Department of Energy contract DE-AC03-76SF00515.

†Present address: University of California, Dept. of Physics, Davis, CA 95616.

SYSTEM OVERVIEW

The system is generalized and database-driven. New feedback loops which behave in a linear fashion are implemented by configuring the database and hardware but without requiring additional software. Furthermore, special-purpose capability is provided to handle non-linear functions such as energy control with phase shifters. The system is based on the state space formalism of digital control theory [3]. Vectors of measurements, states and actuator values are manipulated using matrix and vector arithmetic. Matrices are calculated offline and stored in a database for online use. Measurements input to the feedback system are typically Beam Position Monitor (BPM) readings. The state vector includes calculated quantities such as beam position, angle or energy which the feedback loop controls to user-selected setpoints. Actuators which control the beam are typically analog control devices such as steering dipole magnets, Klystron amplitude controllers and phase shifters.

The major software components of the system are shown in Figure 2. Most of the associated software is written in the C programming language. The SLC database contains device specifications, display information and control parameters associated for all existing feedback loops. Only the software which runs on the VAX has access to the entire database. The VAX software is responsible for initializing and arbitrating feedback processing in the micros and handling user requests. There is an extensive selection of displays available to allow users to monitor and analyze the feedback behavior in addition to facilitating studies of the SLC itself. The VAX software uses an object-oriented architecture. Feedback loops, database-driven displays, and vector elements are among the types of objects which are manipulated in a generalized manner [4].

The micro software executes all of the real-time control functions, including taking measurements, performing calculations and implementing new actuator settings. Since these functions may be distributed across several micros, a specialized high rate network system, KISNet [5], has been adapted from the Advanced Light Source (ALS) project in order to transfer measurement and actuator data between micros. The feedback software which runs on the micros is divided into three functions: measurement, control and actuation. For a single feedback loop there may be multiple measurement and actuator tasks running on different micros with each responsible for its own hardware. A single controller task for each loop receives all of the measurement data, performs calculations, and sends new settings to the actuator task(s).

The matrices used in the controller calculations are determined by an offline simulation program [6] which is based on the MatrixX package from Integrated Sytems Incorporated. The matrices are designed to minimize the RMS of the controlled states, provide good response to step functions, and to maintain stability when the machine response does not exactly match the model. The design involves tradeoffs between quick response and stability under changing beam conditions. The response characteristics may be tuned in the matrix design by adjusting the noise spectrum expected from the accelerator, although in the SLC the same setup is typically used for all loops. The matrices are initialized using the theoretical model of the accelerator. The model is usually good over short distances.

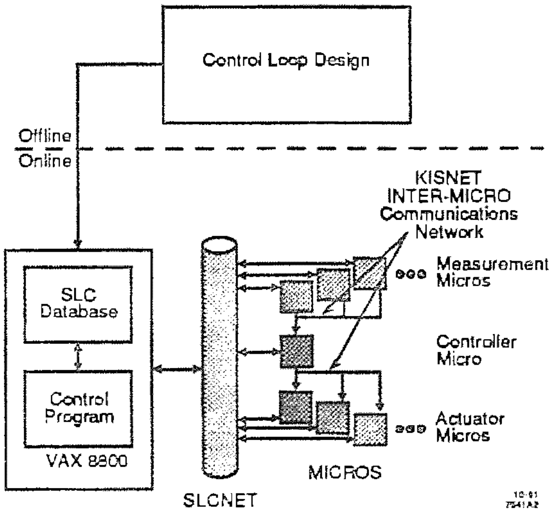


Figure 2: Feedback System Architecture.

The simulation program has been very useful for predicting stability of the feedback processing and determining the workability of new algorithms. Most of the problems encountered in operation were predicted in advance by the simulator, and some potential problems were circumvented by the software. This is one of the reasons that commissioning the feedback loops has been a remarkably smooth and minimally invasive process.

FEEDBACK CALCULATIONS

The feedback algorithm can be summarized in two equations which are based on the predictor-corrector formalism of digital control theory [3]. This algorithm has previously been described elsewhere [6] in further detail. The first controller equation estimates the values of states which are associated with the feedback loop, based on the previous state estimate, currently implemented actuator settings, and measurements.

$$\hat{\mathbf{x}}_{k+1} = \Phi\hat{\mathbf{x}}_k + \Gamma\mathbf{u} + \mathbf{L}(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_k) \quad , \qquad (1)$$

where

$\hat{\mathbf{x}}_k$ is the estimate of the state vector on the k^{th} pulse.

Φ is the system matrix and describes the dynamics of the accelerator model.

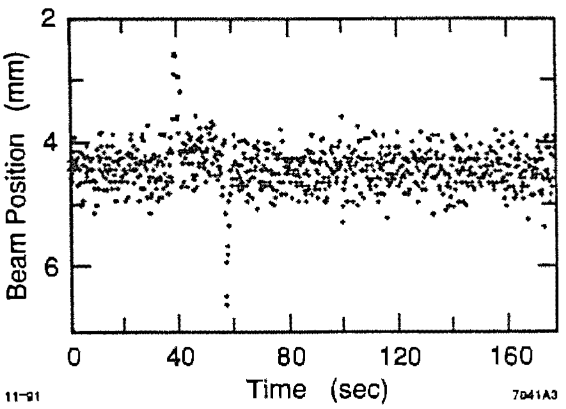


Figure 3: Feedback Response to Step Functions.

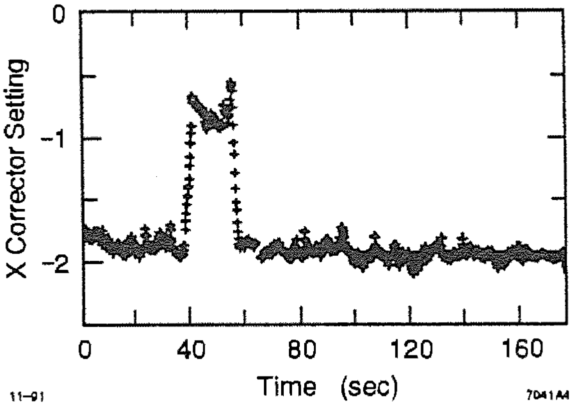


Figure 4: Actuator Control for Step Functions.

Γ is the control input matrix. It describes how changes in the actuators should affect the state.

u is the actuator vector. It contains the current actuator settings with reference values subtracted.

L is the Kalman filter matrix. Given an error on the estimate of the sensor readings, it applies a correction term to the estimate of the state vector.

y is the measurement vector. It contains the current measurements with reference values subtracted.

H is the output matrix. It maps the state vector to the output vector. That is, given an estimate of the states, it gives an estimate of what the sensors should read.

The matrices **Φ**, **Γ**, and **H** are obtained from the model of the accelerator. The **L** matrix is derived from the other matrices and is designed (via the Linear Quadratic Gaussian method) to minimize the RMS error on the estimate of the state.

The second controller equation calculates the actuator settings based on the estimated state vector.

$$\mathbf{u}_{k+1} = \mathbf{K}\hat{\mathbf{x}}_{k+1} + \mathbf{N}\mathbf{r} \qquad (2)$$

where

K is the gain matrix. It is derived in a manner similar to **L**. It is designed to minimize the RMS of selected state vector elements.

N is the controller–reference–input matrix. It maps the reference vector to actuator settings and is directly derivable from the model of the accelerator.

r is the reference vector which contains setpoints for the states controlled by the loop.

DIAGNOSTIC CAPABILITY

The micros save measurement, state and actuator data for the last few hundred iterations in a ring buffer; this data is available for display upon user request. These ring buffer displays are one of the most useful diagnostics of the system, enabling analysis of perturbations and beam losses after they have taken place. If the user requests the display within a minute or so after such an event, the associated data is usually available. This functionality is also useful for studies of beam jitter and other phenomena. Figure 3 shows how a feedback-controlled beam position changes with time. Figure 4 shows the associated corrector values during the same time period. During this period, two step functions were purposely introduced to perturb the beam upstream of the feedback loop in order to test the feedback response; one can see how each perturbation is corrected within several pulses. Typically, the first pulse after a large perturbation is rejected by the feedback filtering software as spurious and then the new state estimate is exponentially averaged over several pulses. Newly calculated settings are usually implemented within one or two pulses for most types of actuators.

Additional analysis capabilities include Fourier transforms and plots of the ring buffer data. The same data may be formatted onto disk files which are compatible with offline analysis packages. Beam orbit plots are available to graphically display the beam trajectory through the range of each feedback loop for comparison with the model-predicted orbit. A history plot capability enables review of feedback control over a period of days, weeks or months. Figure 5 shows how a feedback-controlled beam position differs from its setpoint value over a period of fifteen days. The tolerance lines show that for most of the period shown the feedback loop controlled the beam successfully.

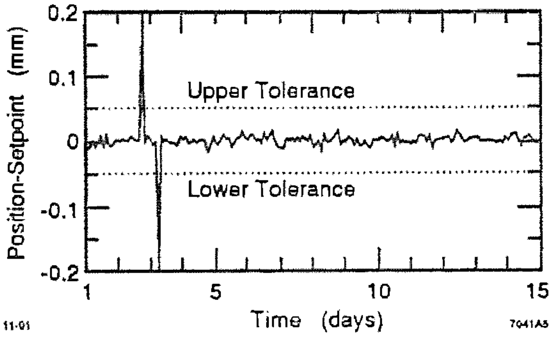


Figure 5: History Plot of Feedback Control.

ADDITIONAL FEATURES

Many system features have been added to the basic algorithm to handle exceptions, improve robustness and add flexibility to the system. For example, a gain factor is implemented as a modification to the second controller equation (actuator calculation), allowing online adjustment of the loop response. The matrices are set up to provide optimal control with a gain factor of 1.0 but it is convenient to allow operators to modify this in response to various operational problems.

An additional capability is the handling of "gold orbits" and loop setpoints. A gold orbit is saved by operators when the state of the accelerator in the region of a feedback loop is believed to be well-tuned. This results in saving the current measurement values and actuator settings. The "gold" measurements are used as offsets in the controller equations. In general the feedback loop will try to control the beam to maintain the gold orbit measurements. The gold orbit values are saved in configuration files which can be reloaded at a later time, facilitating easy reproduction of a particular machine configuration. The operators may wish to tune the machine while the feedback is on by changing the setpoints which control the associated state values. Setpoints may be entered manually, or they may be assigned to a knob and adjusted by turning the knob.

In order to insure that the feedback system does not misbehave, a large part of the micro software involves exception handling. In fact only a small part of the micro software is required for implementing the basic control algorithm. The measurement and actuator values are checked to verify that they are within reasonable limits. If measurements are out of range, have bad status or are not received by the controller, the "expected" values are used in the calculation, based upon the previous state estimates. This allows feedback loops which control both electron and positron beams to continue controlling one beam when the other beam is absent. In order to insure that a single wild pulse does not adversely impact the feedback response, two types of filtering are implemented. Firstly, measurements which vary significantly from the previous pulse are not used unless the value is between that of the two previous pulses. Secondly, an exponential filtering mechanism is built into the matrices to improve stability.

A calibration function is provided for online measurement of how the beam states change with actuator settings. After the calibration is performed, the user may compare the resulting matrices with the model values and with the currently implemented matrices. An option is provided to implement the new values. This function facilitates diagnosis of how well the feedback loop is performing in addition to improving the feedback response when the model is imperfect.

TESTING ENVIRONMENT

In order to test the initial system as well as new developments, a hardware-based feedback test system has been developed. The typical test feedback loop has three measurements, two states and two actuators. The hardware simulator modifies the values of the measurements to respond to changes in the actuators. A function generator introduces variances in the measurements such as sinewaves or step functions. In addition to facilitating debugging and testing of software features without impacting operation of the SLC, the simulator enables the study of feedback response for various system changes.

OPERATIONAL EXPERIENCE

The feedback system was first commissioned in the SLC in November 1990. Since then seventeen feedback loops have been implemented, with more planned. These are shown in Figure 1. At the front end of the machine a new feedback loop will soon control the intensity of the beam from the polarized laser gun. Steering loops control the beam trajectory from the injector into the linac and also into the damping rings. Additional loops steer the beams out of the damping rings, down the LINAC and into the final focus. An energy loop controls the electron and positron beams into the damping rings. A special-purpose calculation is used to control the energy of the electron beam into the positron target. Furthermore, the prototypical systems for energy control at the end of the LINAC and interaction point collision control are scheduled to be replaced by the generalized software in the near future. Additional steering and energy loops are planned for the new Final Focus Test Beam facility.

The number of feedback loops implemented is much larger than originally planned. More loops were added because of the success of the initial loops and the ease of commissioning. The system worked much better than anticipated. Loops which involve a single micro could be added just by setting up database entries, without requiring any additional hardware or software. Most feedback loops were commissioned by turning them on with a small gain factor, gradually turning up the gain, and monitoring the loop response by looking at the feedback displays. One of the feedback loops was commissioned "accidentally" by an operator who didn't realize the loop wasn't ready; he just turned it on and it worked.

Fast feedback has become an important part of the SLC Control System and is heavily relied upon for stabilizing the machine. It is now much easier to reproduce a particular machine state after interruptions. Furthermore, operators who were previously occupied with keeping the beams stable have been freed to work on more subtle aspects of machine tuning. Since the system was implemented, the average rate of steering knob turns has decreased by a factor of five. The LINAC is now steered every few days instead of once a shift. The orbit bumps required to minimize the beam emittance are now stable for weeks instead of hours. There are fewer machine protection trips because the scavenger energy feedback keeps the beams centered. Since the feedback system was implemented, accelerator performance has been greatly improved and much of this improvement is attributable to fast feedback. All of the major operational goals for the SLC in the last running period were met or exceeded.

"CASCADED" FAST FEEDBACK

One problem encountered in operation was predicted by the initial simulations and is a result of the large string of steering loops down the LINAC. In the current system, these loops are all controlling the same parameters, resulting in overcorrection of upstream perturbations and amplification of beam noise. As a temporary measure, the gain factors of several feedback loops have been decreased, but this reduces the system's effectiveness.

An enhancement to the feedback system, called "Cascading", is currently under development. It enables a series of fast feedback loops to communicate, eliminating overcorrection problems and allowing the use of optimal gain factors. In the new system, each upstream loop sends its calculated state vector to the adjacent downstream loop. It is not necessary for an upstream loop to communicate with all loops downstream of it. The downstream loop controls the difference between state elements calculated from downstream beam position monitors and the transported values of the associated states calculated by the upstream feedback loop. This results in each loop correcting only those perturbations not already removed by the upstream loops.

This coordination between feedback loops depends upon a reliable method for mathematically transporting the position and angle at one point to the position and angle at a downstream location. The model of the accelerator, based upon a knowledge of the focusing strengths of the LINAC quadrupoles, provides a basis for this transport, but it is believed that the model is not acceptably accurate over the distances involved. Furthermore, the physical transport characteristics may change during operation. Therefore adaptive methods are used to dynamically update the transport matrices. The adaption calculations are based upon the SEquential Regression (SER) algorithm [7], adapted for use in the SLC feedback system [8].

The adaption is an iterative process which has as inputs the calculated states for a feedback loop and the same states as calculated by an upstream loop. Averaged over time, the transported upstream states should equal the downstream states. The adaption process calculates a transfer matrix which minimizes the difference between the transported upstream and downstream states. This process runs on the same micro as the feedback controller but is implemented as a separate task, allowing the adaption to run more slowly and at a lower priority in order to minimize the CPU impact.

The algorithm is as follows: On each pulse for which the transport matrix is to be updated the following is calculated:

$$\mathbf{S} = \mathbf{Q}(k-1)\mathbf{y_C}(k) \tag{3}$$

$$\gamma = \frac{\alpha}{1-\alpha} + \mathbf{y_C}^T(k)\mathbf{S} \tag{4}$$

$$\mathbf{Q}(k) = \frac{1}{\alpha}\left(\mathbf{Q}(k-1) - \frac{1}{\gamma}\mathbf{S}\mathbf{S}^T\right) \tag{5}$$

where:

$\mathbf{y_C}(k)$ is the state vector from the upstream loop with setpoints subtracted.

k is the beam pulse number.

$\mathbf{Q}$ is the estimate of the inverse of the covariance matrix of $\mathbf{y_C}$.

$\mathbf{S}, \gamma$ are intermediate results.

and

$$\alpha = 2^{-1/\tau} \tag{6}$$

where

τ is the number of pulses for covariance matrix averaging, typically 50.

A large γ means the beam fluctuation has suddenly increased, which could cause the transport calculation to be unstable. Therefore the following equations which update the transport matrix are calculated only if γ is less than a cutoff value, typically 20.

$$\begin{aligned}\epsilon \;&=\; (\text{raw state vector}) - (\text{raw state setpoints}) \\ &\quad -\mathbf{T_C}(k)\mathbf{y_C}(k)\end{aligned} \tag{7}$$

$$\begin{aligned}\mathbf{T}_i(k+1) \;&=\; \mathbf{T_i}(k) + \\ &\quad \eta\mathbf{Q}(k)\mathbf{y_C}(k)\epsilon_i\end{aligned} \tag{8}$$

where

$\mathbf{T}_i$ is the estimate of the i^{th} row of the transport matrix $\mathbf{T_C}$.

η is the learning rate or gain, typically 0.1 or 0.2.

The calculation of $\mathbf{T}_i$ must be evaluated for all i, that is for each row of the transport matrix. If there are changes to the physical model the T and Q matrices converge to new values within a few minutes.

Simulations indicate that this method will behave reliably and will improve the feedback response. It should completely eliminate the overcorrection problems previously experienced. The new system is scheduled to be ready for commissioning by the end of 1991.

CONCLUSIONS

The new fast feedback system has been a remarkably successful addition to the SLC Control System. The generalized approach enables easy addition of new loops and expansion of functionality. Commissioning of new loops has caused relatively little negative operational impact, due to use of simulation and offline testing. The database-driven design and reliance upon existing hardware also helps to minimize commissioning effort. The user interface is easy for operators to use and provides extensive analysis capability. Most importantly, the system has improved the stability and tuning of the SLC, enabling operational goals to be met.

ACKNOWLEDGEMENTS

The authors wish to thank J. Zicker and S. Castillo for their work on the system design. We also appreciate the software contributions of P. Grossberg, R. Hall and L. Patmore.

REFERENCES

[1] G. Abrams et al., “Fast Energy and Energy Spectrum Feedback in the SLC LINAC,” SLAC–PUB–4183, in *Proceedings of the Particle Accelerator Conference*, Stanford Linear Accelerator Center, March 1987.

[2] F. Rouse et al., “Maintaining micron size beams in collision at the Stanford Linear Collider,” SLAC–PUB–5512, in *Proceedings of the IEEE Particle Accelerator Conference*, Stanford Linear Accelerator Center, May 1991.

[3] Gene F. Franklin and J. David Powell, *Digital Control of Dynamic Systems*, Addison–Wesley, 1980.

[4] F. Rouse et al., “Design of VAX Software for a generalized feedback system,” SLAC–PUB–5511, in *Proceedings of the IEEE Particle Accelerator Conference*, Stanford Linear Accelerator Center, May 1991.

[5] K. Krauter and D. Nelson, “SLC’s Adaptation of the ALS High Performance Serial Link,” in *Proceedings of the IEEE Particle Accelerator Conference*, Stanford Linear Accelerator Center, May 1991.

[6] T. Himel, L. Hendrickson, F. Rouse and H. Shoaee, “Use of Digital Control Theory State Space Formalism for Feedback at SLC,” SLAC–PUB–5470, in *Proceedings of the IEEE Particle Accelerator Conference*, Stanford Linear Accelerator Center, May 1991.

[7] Bernard Widrow and Samuel D. Stearns, *Adaptive Signal Processing*, Prentice-Hall, Inc., 1985.

[8] T. Himel, “Requirements and Design Overview for Adaptive Cascaded Feedback,” Internal SLAC document, Stanford Linear Accelerator Center, September 1991.

SMART MACHINE PROTECTION SYSTEM*

S. Clark, D. Nelson, A. Grillo, N. Spencer, D. Hutchinson, J. Olsen
D. Millsom, G. White, T. Gromme, S. Allison, K. Underwood, M. Zelazny, and H. Kang
Stanford Linear Accelerator Center, Stanford University, Stanford, CA 94309, USA

Abstract

A Machine Protection System implemented on the SLC automatically controls the beam repetition rates in the accelerator so that radiation or temperature faults slow the repetition rate to bring the fault within tolerance without shutting down the machine. This process allows the accelerator to aid in the fault diagnostic process, and the protection system automatically restores the beams back to normal rates when the fault is diagnosed and corrected.

The user interface includes facilities to monitor the performance of the system, and track rate limits, faults, and recoveries. There is an edit facility to define the devices to be included in the protection system, along with their set points, limits, and trip points. This set point and limit data is downloaded into the CAMAC modules, and the configuration data is compiled into a logical decision tree for the 68030 processor.

INTRODUCTION

The Stanford Linear Collider includes a number of safety systems that shut down the Collider when unsafe conditions arise. When the Collider shuts down, it becomes difficult to diagnose the cause of the problem. Often it becomes necessary to terminate the startup multiple times before the problem(s) are corrected and safe continuous operation can resume.

Substantially more effective operation would result from a safety system that would report the cause of the fault from the origin of the equipment trip, allowing safe, lower repetition-rate operation to continue, so that the machine can be used to diagnose itself. Automatic return to higher-rate operation after repairs speeds recovery and allows automatic handling of system glitches.

PROPOSED ENGINEERING SOLUTION

A new Machine Protection System (MPS) is being installed in the SLC that will improve machine protection and utilization. The new system will continue to detect unsafe conditions on a pulse- to-pulse basis; however, it will now rate-limit the machine to continue operation at safe levels for diagnostic purposes. This new system utilizes stand-alone array processors to scan the set of fault detectors (radiation, temperature, flows, etc.), making rate limit decisions based on the type and severity of any detected faults, using machine configuration and parameter limit tables developed by machine and radiation physicists.

Facilities have been included to support logging of all machine state changes; the protection system forwards a message to the control room explaining which input signal faulted, and the nature of the fault. These processes allow operators to determine quickly and directly what the problem is/was and what remedial action is required, with a data trail available for later analysis or post-event review.

Beam rate control will be hard wired into the Master Pattern Generator (MPG) and the Injector interlocks used to control the accelerator, so that the machine can be shut down if the expected rates are not properly executed. Failures of sensors or communications failures in sensor processors are treated as if the associated device or included devices where in a worst case failure mode, and appropriate action is taken.

HARDWARE IMPLEMENTATION

The new system will be implemented as a loosely coupled element of the SLC control system, with common facilities on the CAMAC side, new elements built into VME systems, and integrated SLC user interface and applications facilities. As shown in Fig. 1, the system will

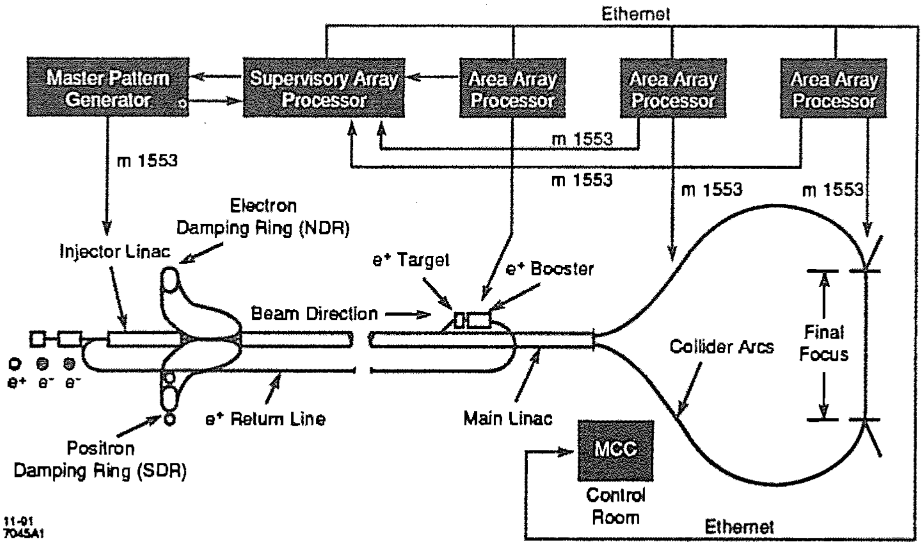
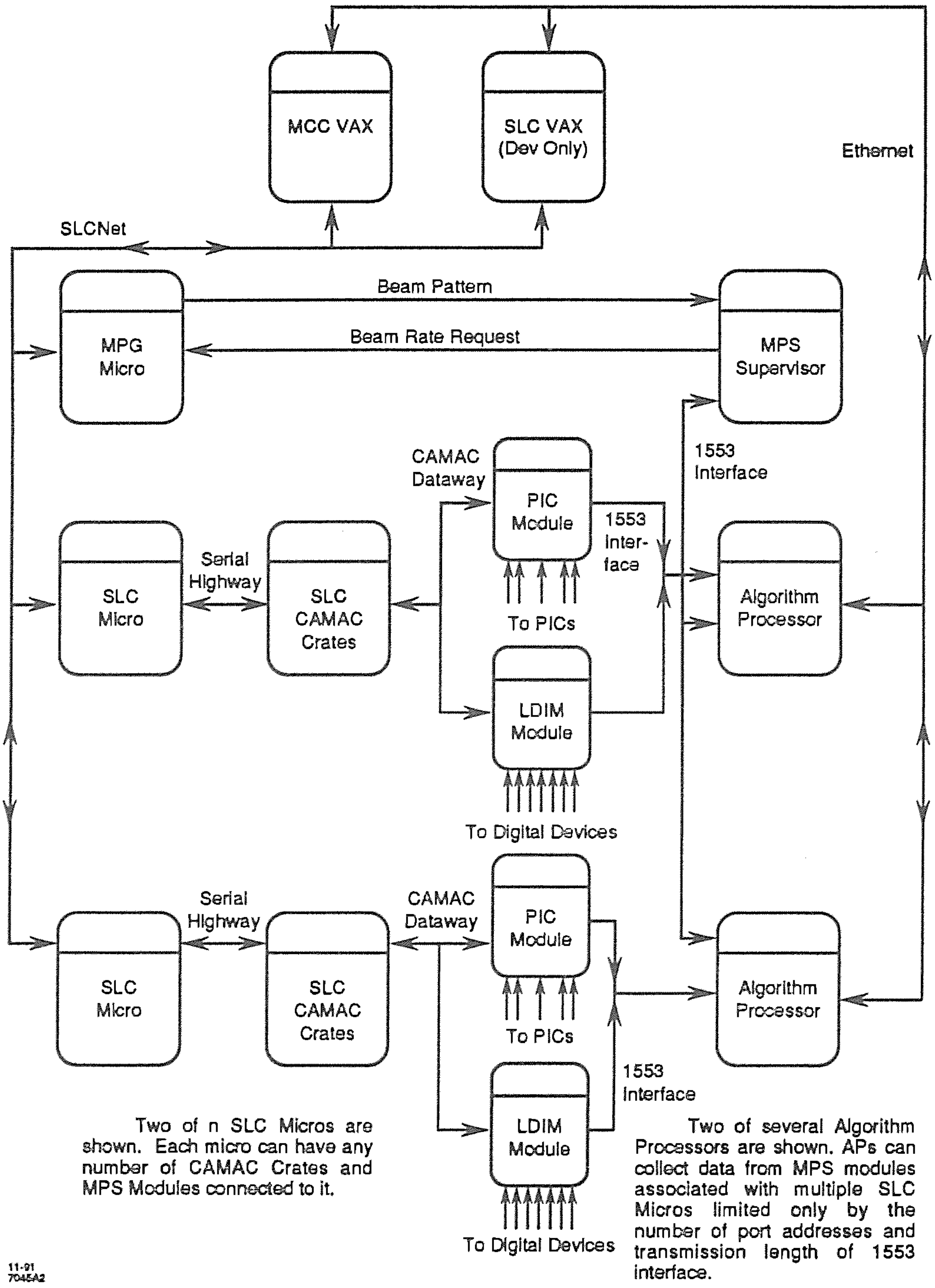


Figure 1.
MPS system architecture.

---

*Work supported by Department of Energy contract DE-AC03-76SF00515.



Two of n SLC Micros are shown. Each micro can have any number of CAMAC Crates and MPS Modules connected to it.

Two of several Algorithm Processors are shown. APs can collect data from MPS modules associated with multiple SLC Micros limited only by the number of port addresses and transmission length of 1553 interface.

Figure 2. MPS computers and network.

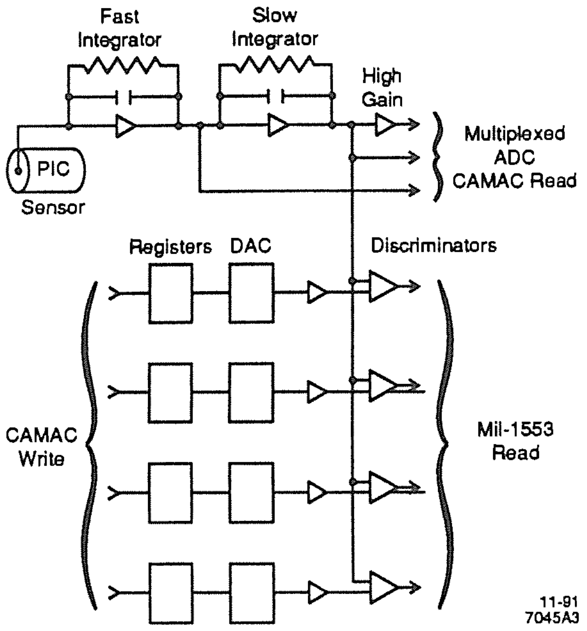


Figure 3. PIC Electronics.

be implemented in clusters structured around functional elements of the accelerator (Linac, Arcs, Final Focus, etc), or around subsystems (injector, positron system, etc.) as appropriate. A modular approach allows new systems to be phased into the overall machine protection system on an incremental basis.

The architecture is modular and consists of CAMAC sensor cards that condition sensor signals and generate go-nogo signals to an Array Processor, which scans these signals and passes on its own summary go-nogo signal to the next level supervisory array processor. The summary process leads to the MPG (see Fig. 2), which sequences the accelerator.

The system is being implemented first in the positron production system, supporting Protection Ion Chambers (PIC's). These Ion Chambers use custom CAMAC modules which interface the radiation sensor and implement the severity level measurements driving the go or no-go signal to the Array processor. Limit levels for the various machine repetition rates are loaded into the PIC from the SLC control system, and scale the acceptable radiation at each beam rate with some hysteresis for stability and automatic return to higher beam rates after the faults are resolved. A facility has been provided via the SLC control system to recover analog measurements from the PIC's for comparison or correlation.

These analog data channels originate in fast and slow integrators in the CAMAC electronics (see Fig. 3), which also drive presettable discriminators that provide the level control signals monitored by the array processors. The system also provides facilities for providing secure changes to level setpoints, and deliberate trips on these control signals to confirm the functionality of the full system and the correctness of the protection system logic.

Future hardware developments will include latched digital status inputs (LDIM), RTD temperature inputs, and Long ion chamber (PLIC). These devices will be set by the SLC control system, and communicate with the array processors via the 1553 bus as do the current PIC's.

System control and data connections are implemented using standard hardware and communication protocols. The real-time control connections between the sensors, the array processors, and the MPG are implemented using MIL STD 1553 as a secure, low- overhead, multidrop communication link (with technology borrowed from Fermilab and CERN). The separate Ethernet link, employing TCP/IP, is used for downloading configuration files, and provides the control room message link for passing the location and nature of any system faults. Both of these links are well-characterized standards, which offer flexibility, reliability, and convenience.

Array Processors (AP) are essentially programmable logic arrays implemented in firmware. These scanners are built from VME 68030 processors running C code on top of a real-time executive. These devices scan the various binary sensor inputs, and evaluate the significance of the signals based on configuration and response tables down-loaded from the VAX via Ethernet. These tables come from a special configuration editor and compiler resident on the VAX. Special security facilities control the integrity of the table transfers, and periodically confirm that no unauthorized or uncommanded changes have occurred in the AP database.

Supervisory Array Processors (SAP) are essentially AP's which combine the outputs of the AP's in the cluster, signal the MPG for beam rate control, and monitor the MPG's responses. This hierarchical model provides for flexibility in grouping MPS sensors, and provides rapid evaluation response by distributing the scan task. A SAP can interrupt the MPG's pipe-lined beam control sequence in 2-3 pulses before accelerator damage can occur. A SAP also has hard-wired access to the injector interlock system should it be necessary to override the MPG.

SOFTWARE IMPLEMENTATION

A configuration editor has been developed so that devices can quickly and accurately be added to the configuration files, using the appropriate boolean operators. This facility provides for the loading of action set-points, device limit parameter, normal device status, and accelerator operation configuration. These files are then compiled into the database for the array processor. The configuration editor then downloads data to the AP's via the Ethernet, and provides a change history and a verification facility to insure system integrity. There is a secure copy of an original or "gold" version of the configuration file which is maintained separately as a global check against all subsequent modifications.

There is a security facility that records equipment bypass actions as they are initiated, and requires that they be reauthorized by senior machine operators at the start of every shift. These listings serve as both a reminder of machine condition and a log of corrective actions yet to be accomplished.

Self-test software is being developed which allows the VAX to test the entire MPS network for continuity, as well as for the correctness of the trip logic. The VAX can initiate trip signals to any selected sensor module and channel, which will be acted upon by the array processors. The test demonstrates a closed loop in the system, and the type of response (limit or shut-down) demonstrates that the logic, limits, and response severity are as specified in the configuration database.

The communication networks were selected for their convenience and for conformance to accepted industry standards. The 1553 bus offers a fast, secure, low-overhead, and multidrop channel for communication between signal conditioning CAMAC cards and the various levels of array processors.

Ethernet with TCP/IP was selected for the message and data links since the messages can be long and there is no requirement for a real-time deterministic type of facility. This tied in nicely with the Ethernet remote debugger purchased with the development environment.

USER INTERFACE

The Control Room User Interface includes facilities to monitor the performance of the system, and track rate limits, faults, and recoveries. Touch panels are used for control, and a series of X-window screens indicate status, faults, and the analysis data from the various applications packages.

SOFTWARE DEVELOPMENT ENVIRONMENT

The MPS Application Software and array processor firmware were developed on the SLC VAX cluster. All software and firmware were developed in Ansi C, and commercially available development tools (such as remote network debuggers, real-time kernel facilities, and standard networks) were used extensively. The microprocessor firmware was developed in a cross-compilation environment and run on the pSOS kernel (Software Components Group).

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S12FC04

FEEDBACK SYSTEMS FOR LOCAL CONTROL OF RACE TRACK MICROTRON RF ACCELERATING SECTIONS

Chepurnov A.S., Gribov I.V., Morozov S.Yu., Shumakov A.V.,Zinoviev S.V.
Institute of Nuclear Physics, Moscow State University
119899, Moscow ,USSR.

Abstract.

In order to obtain an electron beam with an excellent energy resolution and stable characteristics, a tight control of the amplitude and phase of the field in all rf accelerating sections is required. The high rf power level, dissipated in the accelerating section (AS), together with temperature dependence of the AS resonance frequency caused the creation of the original control system of resonance frequency. Amplitude, phase and resonance frequency local feedback control system have been designed. All systems are computer conrtrolled analogue single loops. The control loops guarantee stable, repeatable amplitudes (10^{-3} relative error), phases (+/-0.5°) of the rf fields in AS, resonance frequency of AS (+/-2 kHz) and have optimal bandwidth. A model of feedback loops has been developed that agrees well with measurements.

I. INTRODUCTION.

The control systems of rf power supply system of the accelerating sections of the continuous wave (CW) race-track microtron (RTM) are described in this paper. These systems operate in different parts of the frequency domain and are connected with each other by control parameters. The described systems ensure constant rf parameters of the AS, such as rf power, resonance frequency and phase difference. These systems form the bottom level of the RTM hierarchical computer control system (CCS) [1]. All analog systems are completely controlled by the top level of the CCS trough optocoupled devices. It is possible to change operating modes and reference signals for feedback control systems by an order from the top level of the CCS.

II. RF SYSTEM.

An outline of the rf power supply of the AS, which is a part of the general rf power supply system of RTM, is illustrated in Fig.1. In an operating mode, a reference rf signal of 2450 MHz (RS) passes over a microstrip rf channel to the klystron input port. The output power of the CW klystron is about 25 kW. The RS is stabilized in frequency up to 1 KHz and in power up to +/- 10^{-3}. The klystron is connected to the AS by a waveguide through the circulator, vacuum window and vacuum port. The incident and reflected waves are checked by means of the double directional coupler (DC) and diode detectors D_1 and D_2. A signal from the rf probe, located in the AS rf power input cell, passes through a 4-channel power splitter to the sensors of amplitude, phase difference, and AS resonance frequency: the detector D_3, the phase detectors PD_1 and PD_2, respectively. The voltage controlled microstrip pin-attenuator A_1 and current controlled phase shifter PS_1 are used as the controllers in the local feedback systems. It is possible to select the operating points of the respective phase detectors with the aid of phase shifters PS_2 and PS_3. Adjusting phase shifters are made as microstrip devices in the form of a meander line on the ferrite layer. They are current controlled, but it is possible to set phase shifters once by special bipolar current pulse train due to the hysteresis property of ferrite. Phase shifters are controlled by the order from the CCS with electron module. The module consists of a relay multiplexer, voltage to current converter and single channel DAC. All necessary rf parameters, such as incident, reflected waves and internal rf field, phase shifts, are measured by the CCS through optocoupled ADC. These signals are amplified and normalized by circuits of analogue feedback control systems.

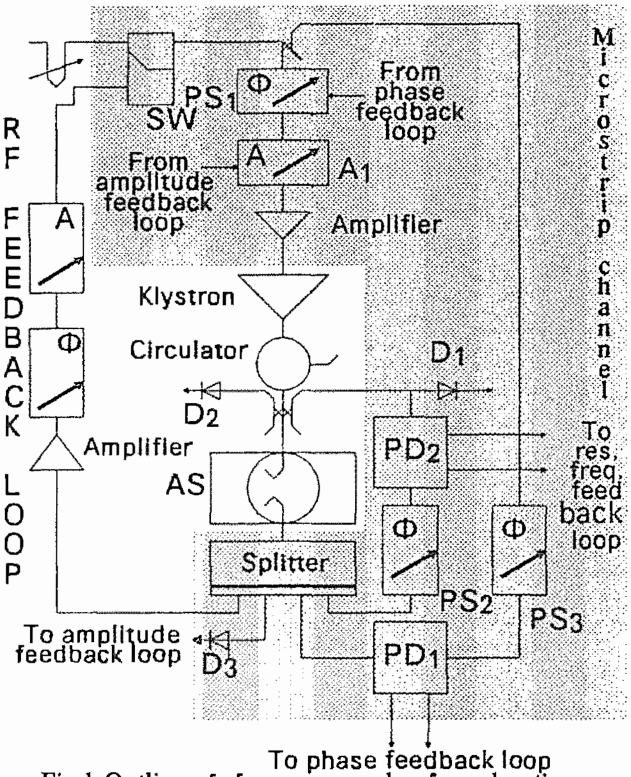


Fig.1 Outline of rf power supply of accelerating section

Mode of power feeding.

The rf feedback loop is closed by the rf switch (SW) in the mode of power feeding into the AS. Constant phase shift and gain of rf feedback loop guarantees amplification of random fluctuations which causes self-excitation of the high power rf field. Frequency of self-excitation is equal to the eigenfrequency of the AS due to the high quality of the accelerating cavity (Q =10 000).

III. RESONANCE FREQUENCY FEEDBACK CONTROL SYSTEM.

The AS resonance frequency is regulated by varying the temperature of water coolant. The resonance frequency decreases due to thermoelastic deformation of the AS and heating of the coolant after the rf power supply system has been switched on :

$$\frac{dF}{dP} = -90\frac{kHz}{kW}\ .$$

Mode of power feeding.

The thermoregulating system (Fig.2) consists of a valve (V) and a voltage controlled thermoelectrical heater (TH) of a total power of 14 kW. The water flow is controlled by valve V and measured by flowmeter FM. The temperature of coolant is measured by two termistors T_1 and T_2, located at the input and output of the AS. The flow relay (FR) checks for the presence of coolant flow and supplies alarm blocking signal. The valve is actuated by a asynchronous motor under computer control . The CCS checks rating value of coolant flow through optocoupled ADC connected to FM.

Digital temperature feedback control system.

In order to feed and support rf power in AS, it is necessary to ensure equality between self-excitation frequency and frequency of the reference signal . The discrete-time computer control system has been designed for temperature tracking . Control system consist of cooling system of the AS, ADC sensors and DAC. It is possible to model the system in the form of "the second-order system with transport delay". Identification has been based on the measurements and computer analysis of the system's responses to a step reference signal input. Control program in the CCS station use both integral and proportional procedures for controlling. System's parameters has been chosen to minimize settling time and overshoot of a step response to a step reference input. The temperature of water is stabilized to an accuracy of 0.02°C. Settling time of a controlling parameter response to a step reference signal is about 100 s.

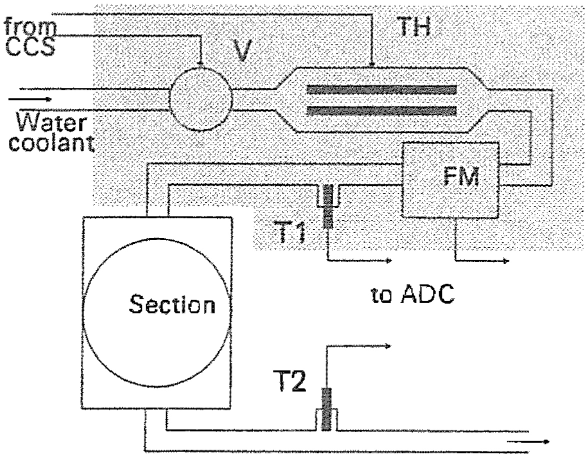


Fig.2 Outline of thermoregulating system of accelerating section.

Analogue resonance frequency feedback control loop.

The feedback loop of analogue feedback control system of resonance frequency is closing after rf power has been fed and external RS excitation has been switched on. At the same time feedback loop of digital temperature control system is opening by the order from the CCS. The error signal for the control system of resonance frequency is supplied by PD_2 (Fig.1) which compares the phases of the AS field and incident field. It is correct due to the relation between phase shift and frequency shift: $F=\frac{Fo}{2Q}tg(\Phi)$ where *F* -frequency shift, *Fo*- resonance frequency, Φ - phase difference, *Q* - quality of cavity. The presence of adjusting phase shifter PS_3 ensures the ability of the sensor to operate in a zero-point, quite linear, domain of static characteristic. The necessary phase shift is set up by the respective adjusting phase shifter by an order from the CCS. When two rf signals with amplitudes of *As* and *Ar* are fed to input ports of PD, two low frequency output signals are expressed as:

$$Vs^2 = As^2 + Ar^2 + 2AsAr\cos(\Phi)$$

$$Vr^2 = -(As^2 + Ar^2 - 2AsAr\cos(\Phi))$$

where Φ- is the phase difference between *Ar* and *As*. Input circuits of analogue control systems consist of operational amplifier (OA) which is summing up two input signals *Vr* and *Vs*. The resulting signal is proportional to the cosine of the phase difference .

We have used PI-compensator circuits with optimized parameters in our system . The object of controlling has been described as a second order system with a transport delay. The model transfer function of the system is:

$$Y(s)=\frac{Ko}{(T1s+1)(T2s+1)}e^{-sTd}\ .$$

The standard parameters of this model , correspond to real objects and are: *T1*= 11.1 s ,*T2*= 3.6 s,*Ko*= 0.067,*Td*= 5.5 s (flow of coolant - 0.4 l/s). At this point, we have used the same model and method as for digital temperature control system to optimize the analogue controller, but in continuous-time domain. The bandwidth of analogue regulator has been limited to 40Hz, because noise fluctuations with higher frequencies of controlling parameter are physically impossible. The analogue regulator consists of a sensor amplifier, a low pass filter, and PI-compensator circuits. We increased gain of open loop. The system remained stable and steady state error decreased due to a low frequency of an up edge of the system's bandwidth. The choice of optimal parameters of the model of PI-compensator circuits has been based upon the desired characteristics of the system: settling time and overshoot of a controlling parameter's step respose. Optimal parameters of the PI-compensator model are: gain K=10.0, integrity time Ti=8.0s, proportional coefficient Kp=1.5. The resonance frequency stability which is provided by this control system, is +/-2 kHz for all real fluctuations in the system.

IV. AMPLITUDE FEEDBACK CONTROL SYSTEM.

In order to ensure the stable rf field in AS, an amplitude control feedback system has been designed. Accuracy of field tracking must be 10^{-3} of relative error. It was difficult to anticipate all possible sources of noise in the system, so the bandwidth has been chosen up to 50 kHz. We have aspired to make the system responce in a closed loop within 10 mks, which is equal to the fill time of AS. These early conditions demanded from designers efforts both in improvement of microstrip pin-attenuators and the creation of original analog circuits with fast operational amplifiers. Furthermore, the electron circuit solves the problem of nonstop operational checking of some important parameters of different accelerator systems, such as the level of the reflected wave, low vacuum, presence of coolant in AS and the klystrons. Information about faults is transmitted to the top level of the CCS and rf inputs of klystrons are closed. Important parameters, on which the safety of the accelerator depends, are monitored twice with different sensors. One of the function of the circuit is to control rf switch, which closes the rf feedback loop. Each element in the feedback loop, from the OA to the voltage controlled attenuator, has been measured to determine ts time responses and frequency-dependent characteristics. Both proportional and integral controls have been used in the feedback control circuits. A prototype of the system has been tested assuming a mini-computer controlled testing panel. The analog model of AS, which consist of OA and RC circuits, has been used. The step responses have been measured with a CAMAC fast ADC (50ns freq. of discret). Random fluctuations of rf amplitude have been measured in real conditions of a working accelerator by using a digital spectrum analyzer. Measurements have been made in closed and open loops. The major noise sources are the high voltage supplies of the klystrons and thermoheater's thyristors with main frequencies of 50 and 150 Hz. Accuracy of the rf amplitude stabilization of about 0.1% of relative error has been achieved. The pin-attenuator is controled by current, which is supplied by an analogue board of the amplitude feedback control system. Static characteristic of the attenuator is quite linear in the operating domain. The operating point has been chosen in the region of 0.5 mA which corresponds to 4 dB of rf attenuation. Internal construction of the microstrip attenuator resulted in an rf phase shift dependence from the controlling current (Fig.3). This "bad" property caused an internal bond of the amplitude and the phase feedback control loop, which was necessary to take into consideration during the later design of feedback loops.

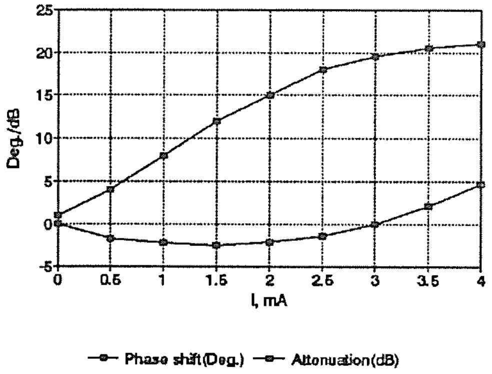


Fig3. Microstrip pin-attenuator performance characteristic.

V. PHASE FEEDBACK CONTROL LOOP.

The phase feedback control loop has been designed at the last stage of complete feedback control system of AS rf parameters designing. Phase control systems guarantee stable phase shifts for every AS, depending on AS location in the linac. PD_1 is used as sensor and PS_1 is used as a controller. An analog phase feedback control system has been created on the basis of the described above analog systems. Sensor circuits is equal to the sensor circuits of the resonance frequency feedback control loop. Regulating circuits are equal to the fast circuits of rf amplitude feedback control loop. The accuracy of the system is about 0.5^{o}.Settling time of a controlling parameter response depends on the type of phase controller. A varactor phase shifter provides a settling time of about 10 mks. A ferrite phase shifter yields 5ms, due to inductance.

VI. CONCLUSION.

The feedback control systems described above were tested , for the first time, during experiments in the capture section of RTM linac [2]. Circuits and methods of designing were improved simultaneously. Models of all systems have been calculated and simulated with the aid of original software. All the systems together with the top level of CCS have ensured stable, safe and comfortable procedure for feeding of power and experimenting with AS.

ACKNOWLEDGEMENT.

The authors would like to express their thanks to professor B.S. Ishkhanov and doctor V.I. Shvedunov who provided enormouse help for the successful designing and testing of the feedback systems.

REFERENCES.

[1] *A.S. Chepurnov, I.V. Gribov, S.Yu. Morozov, A.V. Shumakov and S.V. Zinoviev Moscow University Race-Track Microtron Control System: Ideas and Development. These proceedings.*

[2] *A.S. Alimov, V.G. Gevokyan, I.V. Gribov et al. Beam Acceleration Experiments in the capture section of CW Race-Track Microtron, Preprint NIIYAF MGU, 89-61/138, Moscow, 1989. (in English).*

PLS Beam Position Measurement and Feedback System†

J.Y. Huang, Jinwon Lee, M.K. Park, J.H. Kim and S.C. Won
Pohang Accelerator Laboratory,
POSTECH, P.O.Box 125,
Pohang 790-600, Korea

Abstract

A real-time orbit correction system is proposed for the stabilization of beam orbit and photon beam positions in Pohang Light Source. PLS beam position monitoring system is designed to be VMEbus compatible to fit the real-time digital orbit feedback system. A VMEbus based subsystem control computer, Mil-1553B communication network and 12 BPM/PS machine interface units constitute digital part of the feedback system. With the super-stable PLS correction magnet power supply, powerline frequency noise is almost filtered out and the dominant spectra of beam obtit fluctuations are expected to appear below $15Hz$. Using DSP board in SCC for the computation and using an appropriate compensation circuit for the phase delay by the vacuum chamber, PLS real-time orbit correction system is realizable without changing the basic structure of PLS computer control system[1].

I. Introduction

In an electron storage ring various kind of beam orbit disturbing sources exist, e.g., power line drift and ripple in magnet power supply, magnet and girder deformation by temperature changes, low frequency vibrations from mechanical vibrations of compressors, etc. When these sources are coupled with strong focusing magnets, beam orbit stability is severely deteriorated. Measurement on the spectra of beam position fluctuation shows that the dominant beam position fluctuation appear in the range $0 \sim 100Hz$[1]. In the third generation synchrotron radiation source, stability of the beam orbit is very sensitive to the noise sources. Many beamline users also require very stable photon beam source, i.e., stable within a small fraction of the beam size. Considering the photon beam sizes from Insertion Devices(ID), beam orbit should be controlled within a few μm.

Pohang Light Source(PLS) is designed as the low-emittance synchrotron radiation source[2]. The magnet lattice is $280m$ long, 12-period Triple Band Achromat(TBA) structure. Results of beam dynamics simulations show that dynamic aperture of the circulating beam is much reduced by the closed orbit distortion[3]. Without correction of the orbit distortion, even a single turn orbit may not be closed, i.e., the beam may have no dynamic aperture. In the PLS, effect of all position errors should enter within $150\mu m rms$. For these reasons, a real-time orbit correction system and local beam steering system for each ID beamlines are forseen for the Pohang Light Source.

PLS computer control system has a four-layer hierarchical structure with distributed control computers and communication networks; a host computer for the large scale computation and central database, console computers for the user interface to the control system, subsystem control computers(SCC) and machine interface units(MIU)[4]. Console computers and SCC are connected by Ethernet. SCC and MIU are connected by Mil-1553B data communication network.

PLS Beam Position Monitor(BPM) is designed as VXIbus modules to fit to the digital closed orbit correction system. All the 9 BPM detector electronics and 6 H/V correction magnet power supply(PS) control modules in a lattice period are designed to be VMEbus-compatible and are housed in a single VXIbus crate. Utilizing those VMEbus based BPM system and high performance PLS computer control system, a fully digital orbit feedback system is under development. A dedicated SCC and 12 BPM/PS MIU's constitutes the real-time closed orbit correction system.

There are some practical limitations in realizing the real-time orbit feedback system. Time delays for digital data communication and computation, and phase delay by eddy current effect of the thick aluminum vacuum chamber limit the feedback frequency range below $15Hz$. One of the biggest noise sources from power line ripples is almost filtered out in the design of PLS correction magnet power supply. Therefore, major orbit noises are expected to appear below $15Hz$ in the PLS storage ring.

II. Beam Position Monitoring System

The most important role of the PLS beam diagnostics will be the accurate and fast measurement of beam position for the stabilization of the beam orbit to meet the stringent low emittance lattice design and experimental user requirements. For this purpose, the state of the art beam position monitoring system, featuring measurement

---

[1]† Work supported by MOST and POSCO

accuracy less than 30 μm, wide dynamic range, long time stability, is under development. There are 9 beam position monitors(BPM) in each period which totals 108 BPM's around the 12-period storage ring chamber. Each BPM consists of four button pickup electrodes and signal processing electronics. We designed two types of processing electronics. One of 9 BPM's per period is a wide band detector which has $2M$ position measurement rate to be able to trace the electron orbit turn by turn in the single bunch operation mode. This novel BPM system can be used for the machine study and development as well as for the commissioning of the storage ring. Other BPM's are narrow band processors tuned to $500MHz$ rf frequency for the accurate measurement and correction of the closed orbit.

PLS BPM system should satisfy the following requirements. For the closed orbit measurement; $20\mu m$ resolution in the whole range of operation, $150\mu m$ absolute accuracy including mechanical and thermal errors, life-time orbit stability within the small fraction of beam size at ID chamber and the capability of $15Hz$ real-time closed orbit feedback. Wide band detector electronics should have over $2M$ position measurement rate for the first single turn measurement during the commissioning or turn by turn position measurement in the single bunch operation. This system should also meet the same operational requirement for the closed orbit measurement. During the commissioning, however, $500\mu m$ accuracy would be enough.

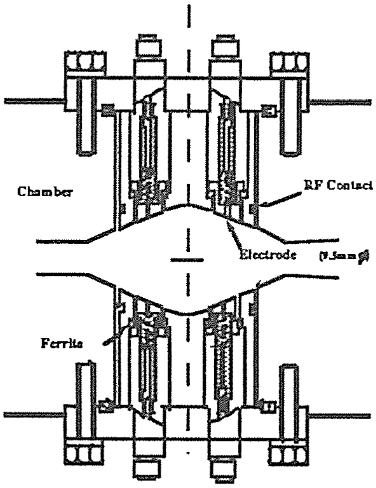


Figure 1: Schematic of the PLS Beam Position Monitor

Since the PLS vacuum chamber is machined from thick aluminum plates, and top and bottom plates are welded together to form the vacuum chamber[5], PLS BPM electrodes are assembled as modular units and mounted on the vacuum chamber with Helicoflex vacuum seals as shown in Figure 1. This modular electrode units have advantage in testing and calibration of BPM modules before installation. In each BPM modules, two electrodes are tightly positioned with ferrite bushings in precisely machined holes. These ferrite bushings damp out various kind of rf resonances as well. Kyocera-SMA feedthrus are welded to be vacuum tight in pairs to the BPM flanges and connected to the electrodes by means of rf spring contacts. In this way, the position of the electrodes will not be affected by the position offset of feedthroughs. The diameter of an electrode is $9.5mm$ which is comparable to the bunch length. The maximum sensitivity in the linear region is about $10\%/mm$. Within the $10mm$ circle, it has shown good linearity. The short bunch signals are picked up by four electrodes and delivered to the electronic detector via coaxial cables. The beam spectrum extends flat to very high frequency with the $3dB$ corner at $9GHz$ and has the high-pass shoulder at $160MHz$, well below the $500MHz$ working frequency. Signal voltage at $500MHz$ is about $22mV$[6].

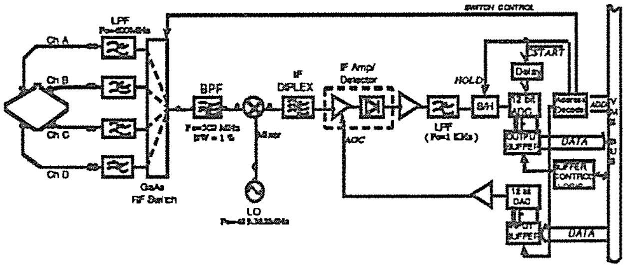


Figure 2: PLS BPM Detector Electronics

A schematic block diagram of BPM detector electronics is shown in Figure 2. Narrow band detector will be used mainly for closed orbit measurement. However this BPM can also be used as the beam-finding tool during the commissioning: by watching whether the beam signal is induced or not on a certain pickup electrode, e.g., button A, we can conclude whether the beam has passed or not. This can also be applied as the first-turn beam position measurement system by four injection beams. By detecting four button signals alternatively induced by four sequentially injected beams, we can measure the first turn beam orbit. There are several narrow band detector systems recently developed, e.g., NSLS[7], ALS[8] and ELETTRA[9] BPM systems. PLS BPM electronics consists of four channel rf switch and a single channel detector. Four pickup signals are scanned via a fast four channel switch and detected in a common processor. After scanning four switches, the fifth clock is used for the detection of the system offset, which is then subtracted from the electrode signals. To avoid the transient periods at both the rising and falling edges of the switching and sampling signal, ADC gate will be set well within the rf switch-on period. Total scanning time is about $100\mu sec$. Fast GaAs rf switch, e.g., SW-254, has low insertion loss, good linearity up to $33dBm$ and good thermal stability; <3% absolute and <0.4% relative drift in the temperature range of $10\sim50°C$. To protect rf switch from the high-power high-frequency components, SLP-600 low pass filter is used before the input to the rf switch. Button signal is again filtered by the band pass filter and mixed down to 10.7MHz with a Vectron CO-233 local oscillator and a SRA-1W mixer. Finally detected signal is digitized by 12bit ADC and interfaced to VMEbus.

A wide band detector will be installed on each period of the lattice and used for both the single turn measurement and the closed orbit measurement. After the first-turn commissioning, most of the fluorescent screens innstalled on ID chambers will be removed when the insertion devices are installed. Wide band detectors are then particularly useful for the trouble-shooting and the recommissionings. A commercially available hybrid junction device, Omni-Spectra Monopulse Comparator Network, will be used as the signal processor. It outputs difference and sum signals from four electrode signals. For the fast turn-by-turn measurement, a 1024-byte FIFO is used to store each sum and difference signals.

With the VMEbus based design of BPM detector boards we have great flexibility in the beam position measurement and feedback operation. One of the novel feature is the capability of broadening the measurement dynamic range by using digitally controllable attenuator in front of the mixer.

III. Closed Orbit Correction System

There are several kinds of closed orbit correction and local beam steering methods: harmonic correction method, least-squared minimization method, the eigenvector method, local bump method, etc[10]. In the Pohang Light Source there are 108 BPM's and 72 horizontal and vertical corrector magnets as the lumped coil windings. Slow drift of closed orbit can be corrected very accurately with these all BPM's and correctors. However real-time correction with those many BPM's and correctors are impossible without using extraordinary hardwares dedicated to the orbit correction system. In the PLS, we want to utilize high-performance PLS computer control system, without changing the basic control system structure, for the real time orbit correction system. Test results show that only a few number of correction magnets and BPM's can suppress the closed orbit distortion to about one-tenth by applying harmonic correction method[11]. Since the harmonic contents of the orbit distortion is dominant near the machine tune value, we can damp out orbit distortion efficiently by using small number of BPM's and correctors.

There are 9 BPM's and 6 H/V corrector magnets in a period of PLS magnet lattice. One section is shown in Figure 3. Two BPM's and a correction magnet in the center of the achromat will be used for the real time orbit correction system, and four H/V correctors and two BPM's at both sides of the ID chamber will be used for the local beam steering system. In ELETTRA, much has been progressed in this kind of real-time orbit correction system[12].

Photon beam position monitors on an ID beamline may also be used as the position detector. However, in the first phase of the PLS construction there will be no insertion devices.Therefore, only computer control system for the local beam steering will be provided without photon beam position monitors. According to the operation mode, the number and locations of correction magnets and BPM's can be flexibly selected.

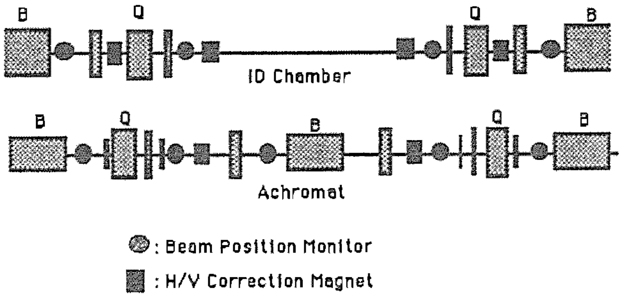


Figure 3: PLS BPM and Corrector Magnet Lattice for Real-Time Orbit Correction

Slow closed orbit correction program will be run at the console computer. Instead, an SCC and 12 BPM/PS MIU's based on the VMEbus and Motorola 68030 microprocessor are dedicated for the real-time closed orbit correction system. The overall structure of the real-time orbit correction system is shown in Figure 4. In each lattice period, a BPM/PS MIU is located for the beam position measurement and correction magnet power supply control. Each BPM/PS MIU crate is also equipped with Motorola 68030 CPU. SCC and MIU's are connected by Mil-1553B data communication network.

Each beam position data read by BPM/PS MIU is transferred to SCC through the serial data communication network, Mil-1553B. In the SCC, harmonics of the orbit distortion are analysed and correction magnet strengths are computed. The correction magnet strengths are then transferred back to twelve BPM/PS MIU's to set the correction magnet power supply currents.

The speed of the real time feedback system is limited by time delays in the control system and phase delays in the magnet power supply, correction magnet and vacuum chamber. Dominant portion of the time delay in the control system is attributed to the serial data communication time through the Mil-1553B field network and large size matrix computations in the SCC. For the matrix computation, a VMEbus based DSP board will be used. To achieve 15Hz realtime feedback frequency with two BPM's and one corrector magnet in each lattice period, total time delay should enter within 8.33$msec$ for the corrections in both horizontal and vertical direction. By analyzing time delays in detail we have got the numbers:

- 100 μsec for the beam position reading
- 110 μsec for the computations in the MIU
- 2400 μsec for the position data transfer from MIU to SCC
- 6600 μsec for the matrix multiplications in SCC
- 1440 μsec for the current data transfer from MIU to SCC
- 4 μsec for the setting of magnet current

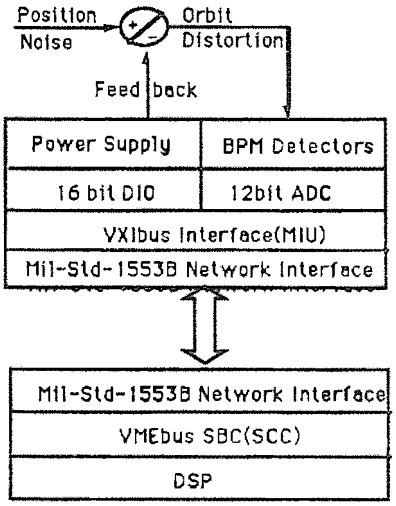


Figure 4: Real-Time Closed Orbit Feedback System

which totals $10.6msec$. This amount of time delay is too large for the purposed feedback system. With a DSP board, computation time in SCC can be reduced to below $600\mu sec$ giving $4.6msec$ time delay in real time orbit control system.

Time delay budget allocated for the corrector magnet power supply, corrector magnet and vacuum chamber is then $3.73msec$. Converting this into the equivalent phase value, we get the phase delay $\phi_D = 40.3°$ at the cutoff frequency $30Hz$. Since the most phase delay will be taken by aluminum chamber because of its eddy current effect, we will try to make the thickness of the correction magnet chamber as thin as possible. Test result conducted in Advanced Photon Source(APS) shows that with an appropriate phase and amplitude compensation circuit, phase delay in power supply + magnet + aluminum vacuum chamber can be reduced to within 40° at $30Hz$[13].

Local beam steering for each ID photon beamline is performed by BPM/PS MIU independent of SCC. Time delays for the beam position reading and computation of the correction magnet current is less than $500\mu sec$. In this case, real-time feedback speed can be extended to the eddy current limited speed of the vacuum chamber. We expect higher than $30Hz$ local orbit feedback speed.

IV. Summary and Conclusion

96 narrow band BPM detectors and 12 wide band BPM detectors are designed for the closed orbit measurement, real-time orbit correction, commissioning and machine studies. All the BPM's are designed to be VMEbus compatible and are housed in VXIbus crates to fit the fast computer control system.

An SCC and 12 BPM/PS MIU's constitute PLS real-time orbit correction system. To achieve $15Hz$ real-time feedback speed, we adopted fully programmable DSP board for the computations in SCC. Total time delay for the digital processes in feedback system is significantly reduced with DSP board. With an appropriate phase compensation circuit for the phase delay by the vacuum chamber, we can realize $15Hz$ global orbit correction system for PLS.

Another sound feature is that the local beam steering system runs in MIU independent of SCC. In this manner, the local beam steering speed depends almost only to vacuum chamber. Higher than $30Hz$ local orbit feedback speed is expected.

References

[1].T.Katsura, Y.Kamiya, K.Haga and T.Mitsuhashi, *A Beam Position Feedback System for Beam Lines at the Photon Factory*, Proc. 1987 IEEE PAC.

[2].Conceptual Design Report, PLS, 1990.

[3].Moohyun Yoon and Taeyeon Lee, *Closed Orbit Distortion and Correction in a Storage Ring*, PLS TR/BD 91-05.

[4].S.C.Won, S.S.Chang, J.Y.Huang, Jinwon Lee, Jae W.Lee, et al. *Status Report on the PLS Control System Development*, presented in this conference.

[5].C.K.Kim, K.H.Gil, W.C.Choi and S.Y.Park, *PLS Vacuum Chamber Design*, J. Korean Vac. Soc., vol.1, 1991.

[6].J.Y.Huang, M.K.Park, E.R.Kim, S.S.Chang and S.C.Won, *Analysis of the Electrical Signal Induced on a Pickup Electrode in the PLS Beam Position Monitoring System*, Proc. 4th Asia Pacific Phys. Conf.

[7].R.Biscardi and J.W.Bittner, *Switched Detector for Beam Position Monitor*, Proc. 1989 IEEE PAC.

[8].J.Hinkson, J.Johnston and I.Ko, *Advanced Light Source Beam Position Monitor*, LBL-25973, 1989.

[9].J.-C.Denard, A.Carniel, R.Aiello and T.Monaci, *Beam Position Monitoring System for ELETTRA*, Proc. 1989 European Accel. Conf.

[10].Alan Jackson, *Closed Orbit Correction in the SRS Storage Ring*, SRS/NS/76/103.

[11].L.H.Yu, *Orbit Stability and Feedback Control in Synchrotron Radiation Rings*, Proc. 1989 IEEE PAC. also see R.J.Nawrocky et al, *Automatic Beam Steering in the NSLS Storage Ring Using Closed Orbit Feedback*, Nucl. Inst. Meth. Phys. Res., A266, 164(1990).

[12].D.Bulfone, *The Real Time Orbit-Position Feedback System for the ELETTRA Storage Ring*, Nucl. Inst. Meth. Phys. Res., A293, 292(1990).

[13].Y.Chung, *Compensation for the Eddy Current, ANL-APS Note*, 1990.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S12FC06

A Position Feedback Control System for the Test Facility of JLC

N.Ishihara
National Laboratory for High Energy Physics(KEK)
S.Nakabayashi, K.Yasuda, K.Ishihara, T.Kubota
Kawasaki Heavy Industries,Ltd

Abstract

In order to develop an alignment system for the Japan Linear Collider(JLC), we have constructed a test facility to study the position control system with multiple degrees of freedom for massive load. Noticeable points of the test facility are as follows.

(1)Feedback fine alignment system which consists of piezoelectric actuators and laser interferometers.
(2)High-speed controller using VME modules.
(3)Level positioner driven by stepping motors.

The controller can easily be connected with other computers by using RS-232C or Ethernet, so that their states such as positions can be monitored by another computer system. This facility achieves the alignment of multi-degrees of freedom with the accuracy of the order of submicron.

I. INTRODUCTION

It is commonly recognized that a submicron alignment system will be required for the final focusing magnets of the future e^+e^- collider. As found in recent study, JLC beams at the interaction point will be as small as 1.4 nm in vertical and 230 nm in horizontal to have enough luminosity [1]. On the other hand, ground motion of the order of 100 nm is expected even at deep underground and the vibration due to the cooling water pulsation is also expected. Therefore we must keep the magnets stable against the vibration and we are considering to realize the magnet position stability by means of a feedback control, called the active alignment [2].

We have constructed a test facility for a 1.5 t magnet. The facility achieves the fine active alignment of five degrees of freedom with piezoelectric actuators. It also has the level positioners of six degrees of freedom as the coarse movers. In this report we will describe the test facility and its control system.

II. TEST FACILITY

The test facility is schematically illustrated in Fig.1 and its photograph is shown in Fig.2. The magnet support table is designed to have enough stiffness (the least natural frequency is above 100 Hz) so that it keeps its own shape unchanged under the usual vibration. The magnet support table is supported by eight piezoelectric actuators (four for vertical and four for horizontal) and the whole active alignment unit is supported by four level positioning units driven by stepping motors.

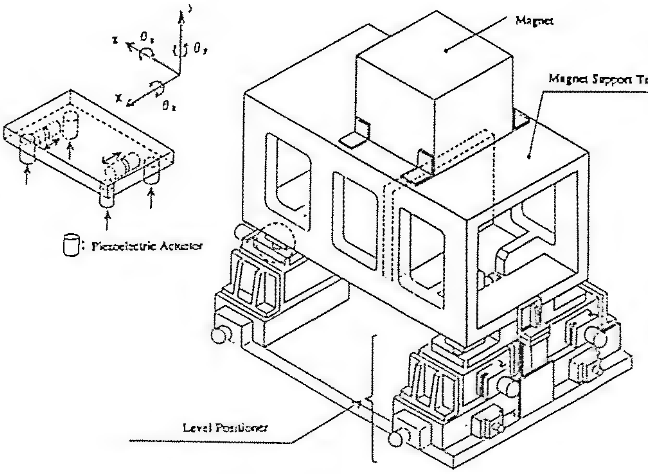


Fig.1 Schematic illustration of the test facility.

Fig.2 Photograph of the test facility.

III. CONTROL STRATEGY AND SYSTEM

Each level positioning unit has a function of three-axis positioning. Each axis has an absolute linear gauge of 1 µm resolution. With a cooperative move of the four level-positioning units, we can bring the table position to within the dynamic range of the piezoelectric actuator.

The fine active alignment of five degrees of freedom, X, Y, θ_X, θ_Y and θ_Z is achieved by the feedback of the table position. We must use a high-speed controller because the fast sampling time and the on-line geometrical calculation are required for the high gain feedback control to suppress the vibration disturbance. The adoption of the VME modules as the controller enables us to use the commercially available boards and the multi CPU structure. Figure 3 shows a schematic illustration of the control system. For the CPU's, we use MC68020 with a clock signal of 20 MHz and obtain a sampling time of 2 ms for the fine active alignment.

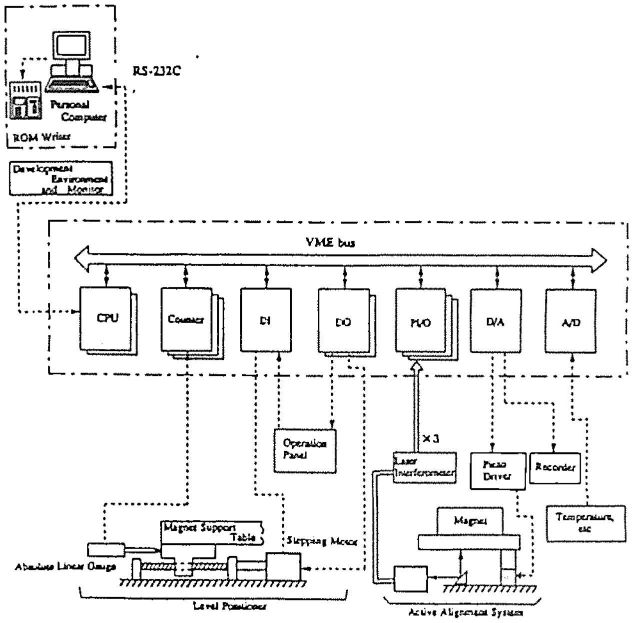
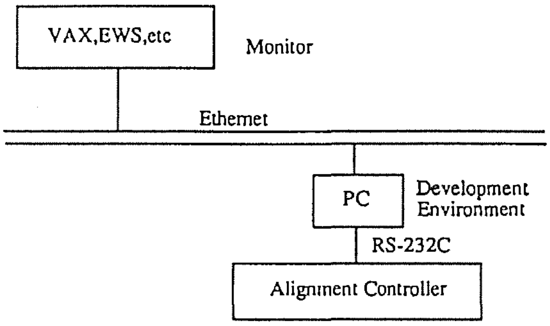
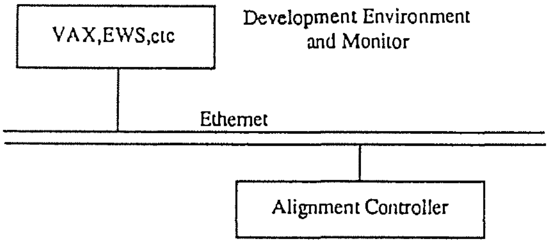


Fig.3 Schematic illustration of the control system.

When we want to connect the control system with a host computer, it is necessary to monitor the data and the status and to change the status by the host computer. We have already made a real-time data monitoring system by using RS-232C as shown in Fig.3. Ethernet is available via a personal computer with this monitoring system. Direct communication between the host computer and this control system with Ethernet is also possible by using the VME module with an Ethernet port and the necessary software. Figure 4 shows some possible network configurations of this VME-based alignment system.



(a)With the personal computer.



(b)With the VME module having an Ethernet port.

Fig.4 Possible network configurations.

IV. TEST

We have tested the precision of the level positioner. The test results are described in rms values of 100 trials as shown in table 1. Positions and tilts of the table have been measured by three laser interferometers in the vertical direction and two capacitance microsensors in the horizontal direction.

As for the fine alignment, we have tested the control performance of three degrees of freedom, Y, θ_X and θ_Z using laser interferometers whose resolution is 10 nm. To investigate the response of the active alignment system to vibration, we have added a white noise with a cutoff frequency of 3 Hz as disturbance to one of the laser interferometer feedback data. The noise has caused imaginary random vibration which simulates the ground motion of Y, θ_X and θ_Z. In Fig.5 showing the response in the time domain, we can see the vibration of amplitude of 1.5 μm peak to peak is damped to less than 0.2 μm.

Table 1 Precision of the level positioner.

	Positioning Accuracy (RMS)
X	0.217 μm
Y	0.750 μm
θ_X	3.00 μrad
θ_Y	2.27 μrad
θ_Z	1.29 μrad

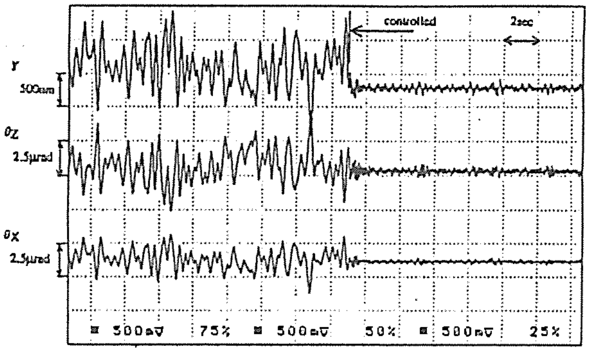


Fig.5 Damping response against the white noise with 3 Hz cutoff frequency.

V. REFERENCES

[1] K.Yokoya, Proceedings of the First Workshop on Japan Linear Collider, KEK, October 1989, pp.26.

[2] N.Ishihara et al., Proceedings of the Second International Workshop on Accelerator Alignment, DESY, September 1990, pp.453.

RF Control System of the HIMAC Synchrotron

M.Kanazawa, K.Sato, A.Itano, K.Noda, E.Takada, M.Sudou, Y.Hirao,
T.Yamagishi*, M.Shigeta*, N.Tsuzuki*, C.Yamazaki*, E.Toyoda*, T.Yagi*, Y.Morii*

National Institute of Radiological Sciences, Acc. Res. Division, 4-9-1 Anagawa, Chiba 260
* Toshiba corporation, 1-1-6 Uchisaiwai-cho, Chiyoda-ku, Tokyo 100

Abstract

An RF control system of the HIMAC synchrotron has been constructed. In this control system we have adopted a digital feed back system with a digital synthesizer (DS). Combining a high power system, performance of the control system have been tested in a factory (Toshiba) with a simulator circuit of the synchrotron oscillation. Following this test, we had beam acceleration test with this control system at TARN-II in INS (Institute for Nuclear Study, University of Tokyo). This paper describes the RF control system and its tested results.

Introduction

HIMAC is a heavy ion accelerator facility dedicated to the medical use, especially for the clinical treatment of tumors. The ion species required for the clinical treatment range from ^{4}He to ^{40}Ar. The required beam energy is from 100MeV/u to 800 MeV/u. This maximum energy is determined so that the silicon ions can penetrate into a human body with a depth of about 30cm. A maximum beam intensity is determined to finish one irradiation within a short time, which is 10^{11}ppp for helium beam. There is also a requirement of low intensity beam of 10^{7}ppp from counter experiment. The HIMAC synchrotron has been designed to satisfy these requirements. In table 1 major parameters of this synchrotron are listed. The characteristic requirements for the RF acceleration system of this synchrotron are followings.

1) Wide RF range (from 1MHz to 8MHz).
2) Wide beam intensity range between 10^{7}ppp and 10^{11}ppp in the synchrotron.

To control wide acceleration frequency stably with low FM noise, a digital control system with a digital synthesizer (Stanford Telecommunication, STEL-1375a) has been adopted for the HIMAC RF control system (See Fig.1). Beam monitors of position (ΔR) and phase ($\Delta\phi$), which can be used with wide beam intensity range, have been developed also. This $\Delta\phi$ monitor must have fast response to use for $\Delta\phi$ feedback loop which damp the synchrotron oscillation. We have checked the $\Delta\phi$ feedback loop with the developed simulator circuit in a factory. In the test with the simulator, we found that the $\Delta\phi$ feedback loop could damp the simulated synchrotron oscillation of frequency up to 6kHz. This result is good enough, because the maximum frequency is 4kHz in the HIMAC synchrotron. As a next step of the test, we have tried to accelerate the beam by use of the developed RF control system.

Table 1
Parameters of the HIMAC synchrotron

Beam species	He^{2+} to Ar^{18+}
Injection energy	6 MeV/u
Momentum spread of the injected beam	<±0.3%
B (injection/maximum)	.1/1.5 T
Field ramp	2 T/sec
Repetition rate	0.5 - 1.5 Hz
Maximum beam energy	800MeV/u
Beam intensity range	10^{7} ppp -10^{11} ppp
Circumference	129.6 m
Transition γ	3.67
Harmonic number	4
Frequency range	1 - 8 MHz
Filling factor	0.8
Peak voltage	<11 kV (at 1MHz)
Synchrotron frequency	1 - 4 kHz

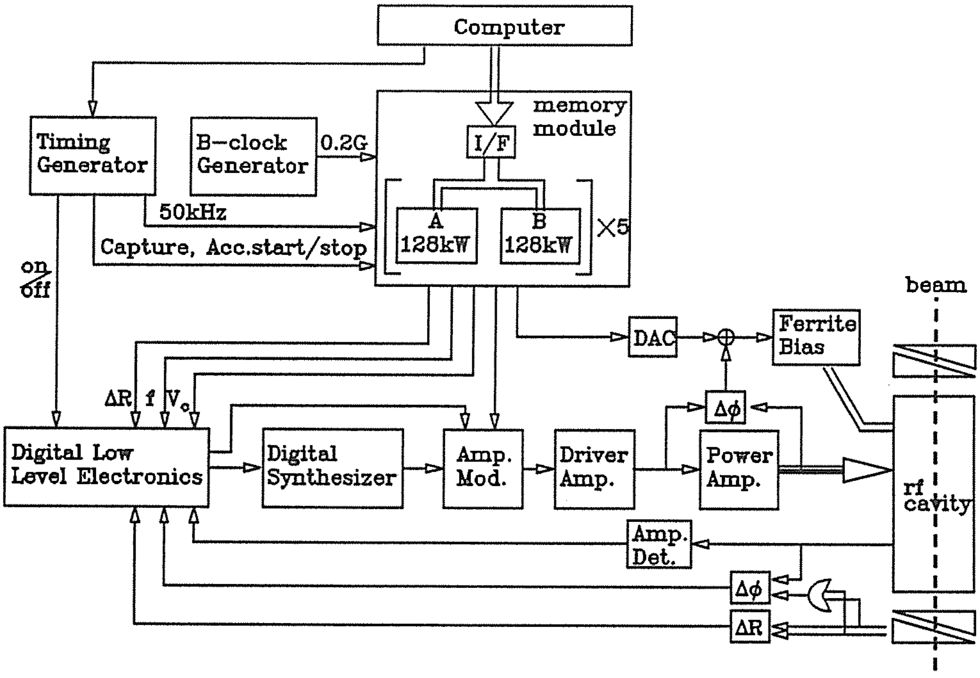


Fig.1 Block diagram of the RF system

Pattern generation

The pattern memory devices are designed to generate the patterns of an accelerating frequency, an accelerating voltage, a ferrite bias current, a bias of beam position, and an accelerating voltage correction. In the computer of man-machine interface, data points of the patterns are given by the formula firstly and can be corrected after by the operator. These pattern data are sent to the computer (RF computer) which control acceleration system directly. The data size of one pattern must be less than 128k words. For each kind of pattern there are two memory units. One memory unit is used to control each device, and the another memory unit is used to change the pattern data in it. With these two memories the pattern data can be changed without stop of the system. This make system tuning easy.

To output the pattern data from the memory to the control devices, two kinds of pulses are used to change the pointer address. One is clock pulse of 50 kHz (T-clock) and the another is the pulse of increment (or decrement) of the dipole magnetic field (B-clock). One pulse of the B^+-clock (B^--clock) corresponds to 0.2 Gauss increment (decrement). To generate the pattern data with these clocks, pattern memories are divided into following three regions.

1) region-1
   T-clock is used to generate the pattern data, which determine the pattern in the flat base.
2) region-2
   B-clock is used to generate the pattern of the accelerating period.
3) region-3
   T-clock is used again to generate the pattern in the flat top.

An event pulse of the beam capture starts to advance the pattern memory address of region-1 with T-clock. To start the beam acceleration, an event pulse changes the T-clock to B-clock and jump to the memory address in region-2 where the output frequency data is same as the last data in the memory region-1. To stop the beam acceleration in the flat top, an event pulse of the flat top changes the memory address to the first address of the region-3. If there is difference between the last output data and the data of the head address in region-3, the output value is moved to the data of the head address with one bit step. This function make the change of the output value smooth and is important not to loss the beam.

Beam monitors

The electrostatic pick-up beam monitors of position (ΔR) and phase ($\Delta\phi$) have been developed, which are described in the other paper[1]. In the following only specific features are listed:

1) The developed monitors can be used at the lowest beam intensity of the HIMAC synchrotron (10^7ppp), and the output errors of the $\Delta\phi$ and ΔR monitors are ±3 deg. and ±3mm with the lowest beam intensity, respectively. The monitor gains can be selected from 0dB to 100dB with 10dB step.
2) Response of the both monitors are fast enough to use for the feed back loops. The delay times of $\Delta\phi$ and ΔR monitors for step function are 7μs and 20μs, respectively.

Digital control circuit

To control the wide range accelerating frequency stably, we have adopted the digital control circuit which is shown in Fig.2. The ΔR and $\Delta\phi$ analog signals are converted to digital data every 2μs with eleven bits (+ sign bit) ADCs which are located near the rf cavity together with the beam monitor electronics. One bit corresponds to 2.4mV (0.052mm in the ΔR monitor and 0.087° in the $\Delta\phi$ monitor). In the control module, the ΔR digital data is biased firstly to control the beam center at optimum position. Then the start-stop function of the feed back with time constant of about 100μs are followed. The ΔR and the $\Delta\phi$ data are added after adjustments of a proportional loop gain in the $\Delta\phi$ feed back, and proportional and integral loop gains in the ΔR feed back. This sum data is further added to the frequency pattern data from the memory module. The data of twenty bits is used to drive the DS, and one bit corresponds to 10Hz. Clock frequency of 10MHz is used for the digital processing, and time delay between digitizing the analog data (ΔR and $\Delta\phi$) and driving the DS with new data is 2μs.

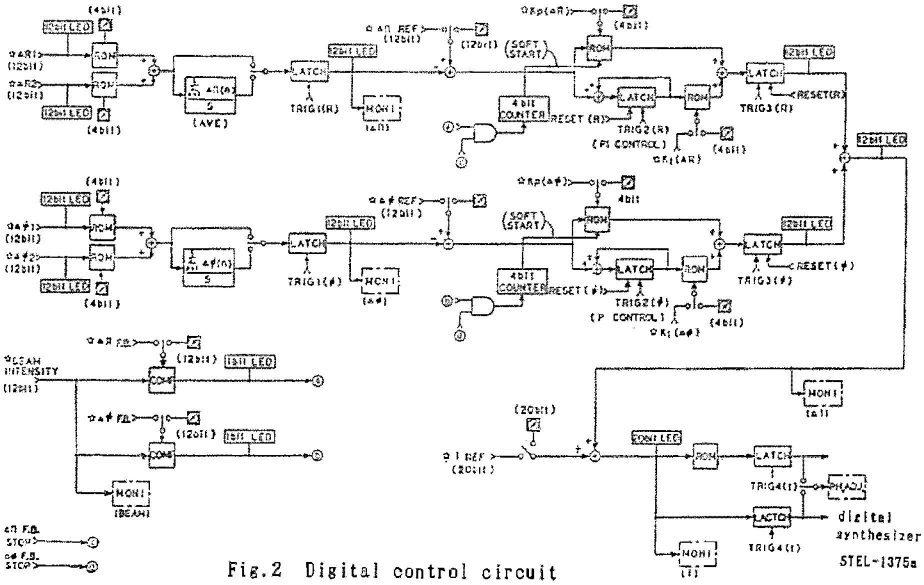


Fig.2 Digital control circuit

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S12FC07

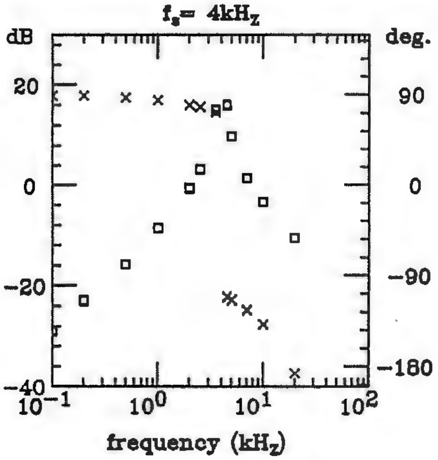


Fig.3 Calculated open loop transfer function of $\Delta\phi$ feed back loop. □;gain, ×;phase.

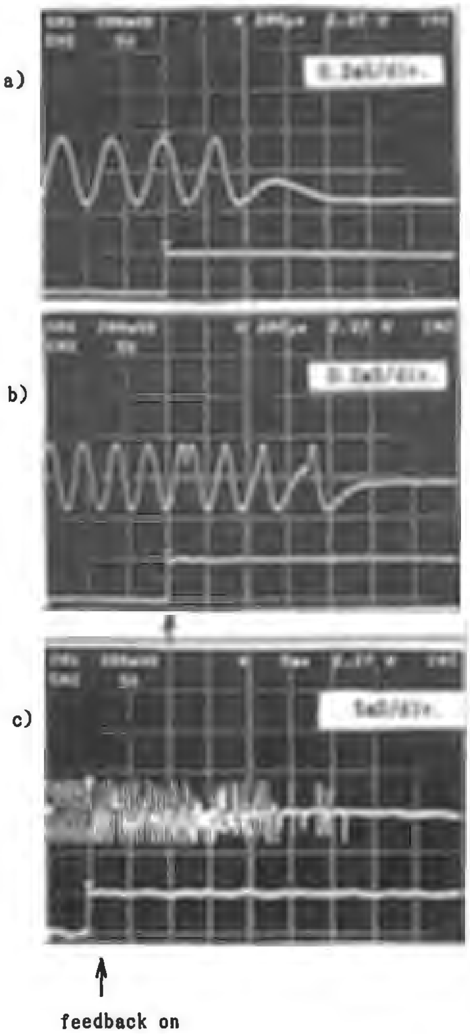


Fig.4 Damping test of the synchrotron oscillation with $\Delta\phi$ feed back with the simulator circuit. The test were performed with synchrotron oscillation frequencies of a) 4kHz, b) 6kHz, and c) 7kHz. The input signal voltage at a first FET amplifier is 4mV(p-p) and f_{rf}=1MHz.

Test of the feed back loop with the simulator circuit

To check the effectiveness of the $\Delta\phi$ feed back loop, an open loop transfer function has been calculated. In this calculation response delays of the following parts have been considered.

1) 7μs in the $\Delta\phi$ beam monitor.

2) 1μs and 2μs for the digitizing the analog data and the digital processing, respectively.

3) 1.5μs in the DS.

4) 1.5μs in the RF cavity.

The calculated result with the critical damping condition is shown in Fig.3. From this result the response of the $\Delta\phi$ feed back is fast enough to damp the synchrotron oscillation of 4kHz, which is maximum frequency in HIMAC synchrotron.

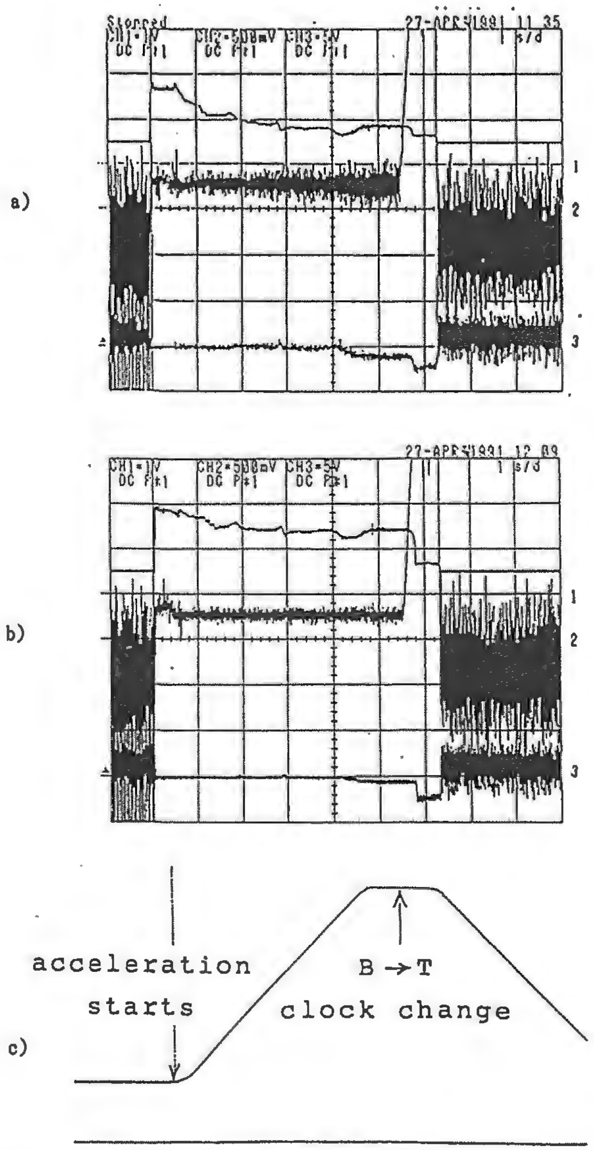


Fig.5. Outputs of the beam monitors. Upper trace is the beam intensity, middle trace is ΔR, and lower trace is $\Delta\phi$ in each picture. Strength of the $\Delta\phi$ feed back loop is a) critical damping b) five times of a). Fig.5-c) shows a corresponding field pattern of the dipole magnet. The acceleration period was 4.46 sec.

To test the Δφ feed back loop further, we have used a newly developed simulator which is a circuit to generated the synchrotron like oscillation. It has the similar transfer function as the synchrotron oscillation. The test was performed with the condition that f_{rf}=1 MHz and f_s (frequency of the synchrotron oscillation) = 4, 6, and 7kHz. As shown in Fig.4, the feed back loop could damp the oscillation of 6kHz quickly. From these results, it is clear that the Δφ feed back loop has fast response which is good enough for our purpose, because the maximum frequency of the synchrotron oscillation is 4 kHz.

Test of beam acceleration

If there is a large phase jump when the accelerating frequency is swept, the accelerated beam will be lost. To make sure that there is no large phase jump in our DS, we have performed the beam test. To do the beam test at TARN-II, we have joined our RF control system (computer, timing generater, memory module, digital low level electronics, digital synthesizer, beam monitor electronics) with the RF system of TARN-II2),3). In this beam test, we could not prepare our B-clock generator of 0.2 Gauss clock. Though the B clock of TARN-II is generated every 1 Gauss increment of the dipole magnetic field, this B-clock was used in the beam test. To see the effect of the corresponding frequency step in the beam acceleration, we have tried to accelerate the beam without the filter which smoothes the step function in the analog RF control system of TARN-II. Though there existed a clear effect of the step function, the beam could be accelerated. We have decided to use the 1 Gauss clock for our digital control system, which has no element to smooth the frequency step in this digital system.

Firstly in the beam test, we have adjusted the position bias in the ΔR feedback loop. If the adjustment was not correct, the beam was lost when the ΔR feedback loop was turned on. The next step was to adjust the strength of the feedback loop on ΔR and Δφ. As the beam intensity was low, the low pass filter of 1kHz was inserted in the ΔR position monitor. From this fact the strength of the ΔR feedback loop has been set at weak value. About the Δφ feedback, the strength of the critical damping condition was not enough. As shown in Fig.5, the best strength of the Δφ feedback loop was about five times stronger than the strength of the critical damping condition at the flat base.

Conclusion

We have constructed the RF control system with the digital synthesizer. The Δφ feed back loop has been checked successfully with the simulator circuit in the factory. In the beam test, we could accelerate the He^{2+} beam from 10MeV/u to 160MeV/u. These energies correspond to the acceleration frequencies of 1.1 MHz and 4.0 MHz, respectively. The tuning of the control system was very simple and easy to succeed in the beam acceleration. This is the characteristic feature of the digital control system and important in the medical accelerator. The acceleration efficiency was about 55%. The possible reasons of the beam loss are;

1) The low beam intensity which makes large noise in the beam monitors, which is due to white noise in the FET of the first amplifier.
2) The RF noise whose effect was enlarged with the low beam intensity.
3) The B-clock step of 1 Gauss which makes the longitudinal beam emittance growth. (In the analog system, we can use the filter to smooth this B-clock step.)

Improving these things with the longer beam monitor electrode and the B-clock step of 0.2 Gauss in the HIMAC RF system, it will be possible to accelerate the beam without large beam loss.

Acknowledgements

We would like to express our gratitude to Dr.Y.Irie, Dr.S.Ninomiya, and the members of the Division of Accelerator Research of NIRS for their support. We also wish to thank Dr. T.Katayama, Mr.T.Watanabe and the members of Accelerator Research Division of INS for their strong support in operation of the TARN-II.

Reference

1) M.Kanazawa et al., "Beam Monitors for RF Control in the Synchrotron TARN-II", Proc. of the 7th Symp. on Acc. Science and Technology, (1989) 210.
2) K.Sato et al,"RF Accelerating System for TARN-II", 11th Int. Conf. on Cyclotrons and their Application, (1987) 333.
3) T.Katayama et al., "RF Acceleration System for TARN-II", Proc. of 7th Symp. on Acc. Science and Technology, (1989) 80.

Development of a VME Multi-processor System for Plasma Control at the JT-60 Upgrade

M. Takahashi, K. Kurihara, Y. Kawamata, H. Akasaka and T. Kimura
Naka Fusion Research Establishment
Japan Atomic Energy Research Institute
801-1 Mukohyama, Naka-machi, Naka-gun, Ibaraki-ken 311-01 Japan

Abstract

Design and initial operation results are reported of a VME multi-processor system [1] for plasma control at a large fusion device named "the JT-60 Upgrade" utilizing three 32-bit MC88100 based RISC computers and VME components. Development of the system was stimulated by faster and more accurate computation requirements for the plasma position and current control. The RISC computers operate at 25 MHz along with two cashe memories named MC88200. We newly developed VME bus modules of up/down counter, analog-to-digital converter and clock pulse generator for measuring magnetic field and coil current and for synchronizing the processing in the three RISCs and direct digital controllers (DDCs) of magnet power supplies. We also evaluated that the speed of the data transfer between the VME bus system and the DDCs through CAMAC highways satisfies the above requirements. In the initial operation of the JT-60 upgrade, it has been proved that the VME multi-processor system well controls the plasma position and current with a sampling period of 250 µsec and a delay of 500 µsec.

1. INTRODUCTION

In the JT-60 Upgrade (JT-60U) [2] where is performed the study of magnetically confined plasma near the thermal break-even condition, the plasma current is increased up to 6 MA in the lower X-point divertor configuration. The vacuum vessel and the poloidal field coils, then, have been replaced for these improvements.

From the viewpoint of plasma equilibrium control, the vertical positional stability is one of the most important issues for the tokamak with elongated plasma. The stabilizing index n_S due to the horizontal magnetic field coil is designed to be 1.6 for the plasma with the poloidal beta β_p=0.6. The vacuum vessel, however, dose not have much effect on the stabilization, because the vessel is made of corrugated thin walls whose time constant of the field penetration is very short (τ=8 msec). Hence, it is necessary to raise the response of the feedback control system. The control cycle of the system must be less than 0.5 msec and the delay of the system must be less than about 1 msec except for the conversion time in the magnet power supplies.

Moreover, since the stored energy of plasma and electromagnetic energy of coils will increase, undesirable events such as plasma disruption may do fatal damage to the components of the vacuum vessel and the coils. More calculations, hence, are necessary to obtain the plasma parameters of positions and clearances more precisely, to produce stable plasmas and to protect the tokamak components. As shown in Table 1, the control system must, then, have such a fast data input/output capacity that it can utilize several tens of status data and several control commands. The control system must also possess such large amount of data transfer capability that it can ship up result data of a few megabytes to its supervisory computer named "discharge control computer" within a limited short time for data analysis at a shot-interval of 10 to 15 minutes.

This paper reports how we designed the control system utilizing VME components in order to satisfy the above requirements for the JT-60U plasma control. Section 2 of this paper describes the configuration of the VME plasma control system. The characteristics of the VME system including its plasma control performance are described in section 3. The final section is a summary.

Table 1 JT-60U Plasma Control Data

Item	No. of Data Channels	Data Amount (kByte)
Input Data		
Magnetic Sensor Signals	70	2,000
Coil Voltages and Currents	11	330
Control References	5	150
Calculation Data		
State Parameters of Position	5	300
Output Data		
Control Commands	5	300
Total	96	2,280

2. CONFIGURATION OF THE VME MULTI-PROCESSOR SYSTEM

As shown in Fig. 1, the JT-60 plasma control system contains two feedback loops. The major loop is for plasma heating and gas fueling control and the minor loop is for plasma position and current control which is done by using five sets of poloidal field coils. Control cycle of each loop was decided corresponding to time scale of change in its control objectives.

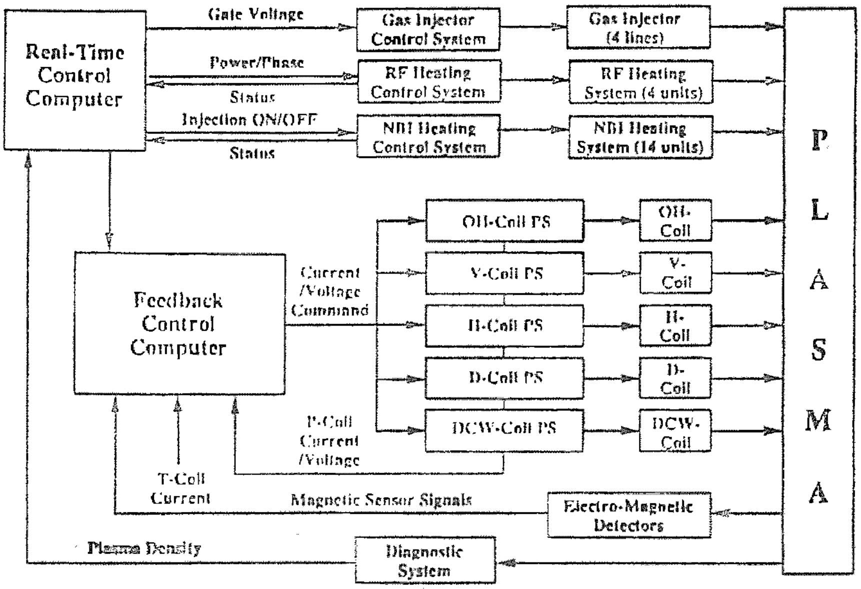


Fig. 1 Data flow in the JT-60U plasma control system

The response of the plasma control system for the JT-60 Upgrade must be more than three times faster than that of the original system from the viewpoint of the vertical positional stability of the plasma. However, it is very difficult to reduce further the execution time required for the data input/output and more accurate calculation because the original JT-60 plasma control system, where the minicomputers and the microprocessor were equipped with accuracy of 16 bit and clocks of 2 to 2.5 MHz, was designed about ten years ago.

Although adoption of analog controllers may be one of the methods to satisfy the above requirements from the viewpoint of control response, it is hard work to keep the controllers be in good condition and it takes much time to develop their control algorithm for sophisticated control.

A VME bus system has the features that (1) we can utilize the fastest processor with 32-bit or more accuracy at present, (2) its system clock (16 MHz) is faster than the CAMAC system clock (5 Hz at maximum), (3) VME bus modules interfacing with CAMAC systems are on the market, etc.. Hence, we have decided to replace the 16-bit minicomputer and microprocessor based system with a VME multi-processor system where 32-bit reduced instruction set computers (RISCs) of MC88100 and 32-bit microcomputers MC68030 are adopted for the plasma feedback control and magnet coil current control respectively.

As shown in Fig. 2, the VME multi-processor system is composed of 4 VME racks, which are connected with each other through 6 bus repeater/expanders (PT-VME902A-1, Performance Technologies, Inc., U.S.A.). One of the racks is dedicated for microcomputers and the others for I/Os. The first slot in the rack A is occupied by a system controller named MVME050A (Motorola, Inc., U.S.A.), which has functions of bus arbitration, system reset, system clock generation and serial clock generation. A host computer of the workstation Sun3/140M (Sun Microsystems, Inc., U.S.A.) is provided for developing the VME microcomputer programs under a Unix operating system, where C language is available.

Three MC88100 based RISCs named MVME181 (Motorola, Inc., U.S.A.) are equipped in the rack A. The 32-bit RISC computer, which supports floating arithmetic, operates at 25 MHz along with two 16-kilobyte cashe memories named MC88200. One of those (CPU#1) is dedicated for collection of the magnetic probe signals and coil currents, calculation of the state variables of plasma current and position, and the plasma vertical position control. The others (CPU#2 and CPU#3) execute the feedback control of plasma current, horizontal position and the height of X-point from the divertor plate, and the calculations for the protective interlock of the coil system.

The rack A is also equipped with a MC68030 based microcomputer named MVME147 (Motorola, Inc., U.S.A.), which communicates with the three RISCs through VME bus and with the host computer through a local area network of Ethernet. The communication programs are executed under a real-time operating system named VxWorks (Wind River System, Inc., U.S.A.).

In the racks B, C and D, analog-to-digital converters (ADCs) and up-down counters (UDCs), which have been newly developed, are provided for digitizing the coil currents and integrating the magnetic probe signals respectively. The ADC has eight input channels and its resolution and conversion time are 12 bits and 5 μsec respectively. The UDC has four 16-bit up-down counters for the input of the pulse signals with a maximum frequency of 2 MHz from voltage to frequency converters. A digital input/output board named MVME340A (Motorola, Inc., U.S.A.) is equipped in order that event signals from the plant and plasma can interrupt the computers and that the computers can put out interlock signals to the actuators. A digital-to-analog converter named DT1403-4 (Data Translation, Inc., U.S.A.) is used for feeding the plasma current to the DDC.

A CAMAC crate is provided for transferring data between the RISCs in the VME system and its supervisory control computers and between the RISCs and the DDCs. The VME system is connected through a CAMAC branch driver named CBD8210 (Creative Electronic Systems, Switzerland), a branch highway and a type A2 crate controller (CCA2,

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S12FC08

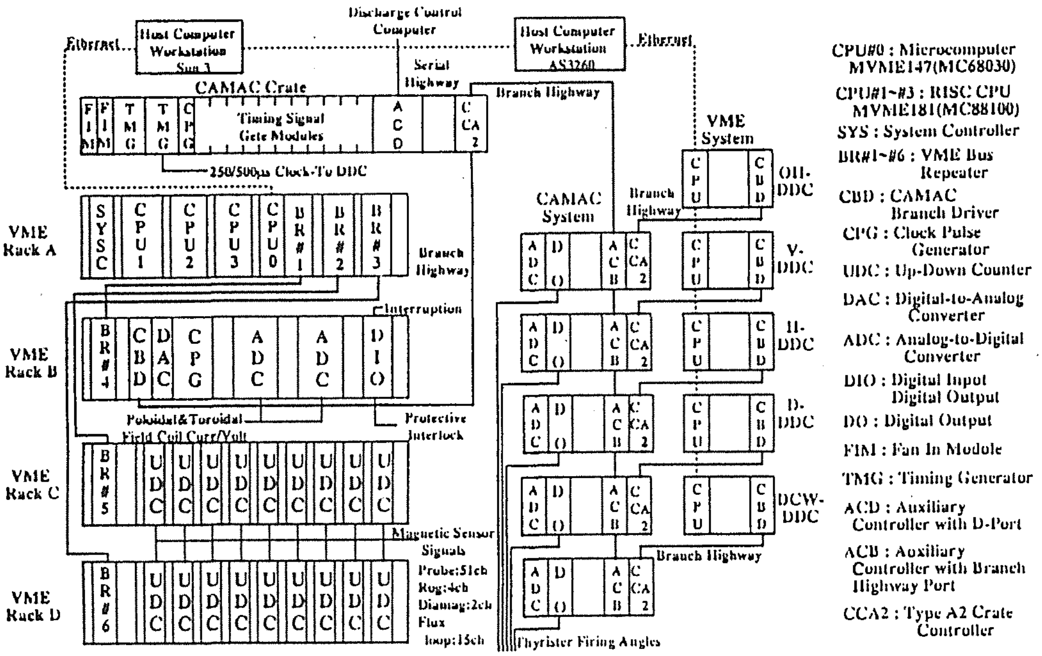


Fig. 2 VME multi-processor system for the JT-60U plasma control system

Standard Engineering Corp., U.S.A.). All the CAMAC system parameters such as addresses, functions and data word length are mapped on the VME address field in the branch driver, which also has a multi crate addressing capability. The branch driver is, hence, suitable for fast and large amount of data transfer in the plasma control.

The CAMAC crate is equipped with an auxiliary controller with D-port (ACD, Kokusai Electric Co., Ltd., Japan), which is provided for transferring the discharge result data from the VME system to the discharge control computer. The control commands are directly transferred to the DDCs through auxiliary controllers with branch highway port (ACB, Kokusai Electric Co., Ltd., Japan). Many timing modules are also installed in the CAMAC crate for generating, receiving, transmitting and masking clock pulses and trigger signals.

3. SYSTEM CHARACTERISTICS

3. 1 Pipeline processing

Figure 3 shows the plasma control time chart of the VME system including the DDC control system. The VME multi-processor system for the JT-60U control is a pipe-lined system with two kinds of sampling clock of 250 μsec and 500 μsec. The former is for the control of the plasma vertical position and the latter for the control of the other parameters.

The CPU#1 first collects the data from the magnetic sensors and the magnet coil power supply shown in Table 1 with a sampling period of 250 μsec. The time required for the data collection is less than 50 μsec. The CPU#1, then, calculates the plasma state variables with the signals from magnetic probes and flux loops. The CPU#1 also executes the calculation for the plasma vertical control and transfers the calculated command of the current or voltage to the DDC in the horizontal field coil power supply (H-DDC) with a cycle of 250 μsec. The time required for the above execution is less than 250 μsec.

The CPU#2 and the CPU#3 execute the calculation for the control of the other parameters such as the plasma current, the horizontal position and the X-point position and the interlock calculation for protecting the magnet coils and the first wall components, following which they transfer their control commands to the DDCs except the H-DDC every 500 μsec. The time required for the execution in these CPUs is less than about 300 μsec. Since the DDCs execute their control calculations in 250-500 μsec, the delay in the plasma control system for the vertical position control is less than 500 μsec and the delay for the other parameter control is less than 800 μsec. The plasma vertical position control loop including the magnet power supply and the magnetic measurement system, hence, has a sampling period of 250 μsec and a delay of about 2 msec in total.

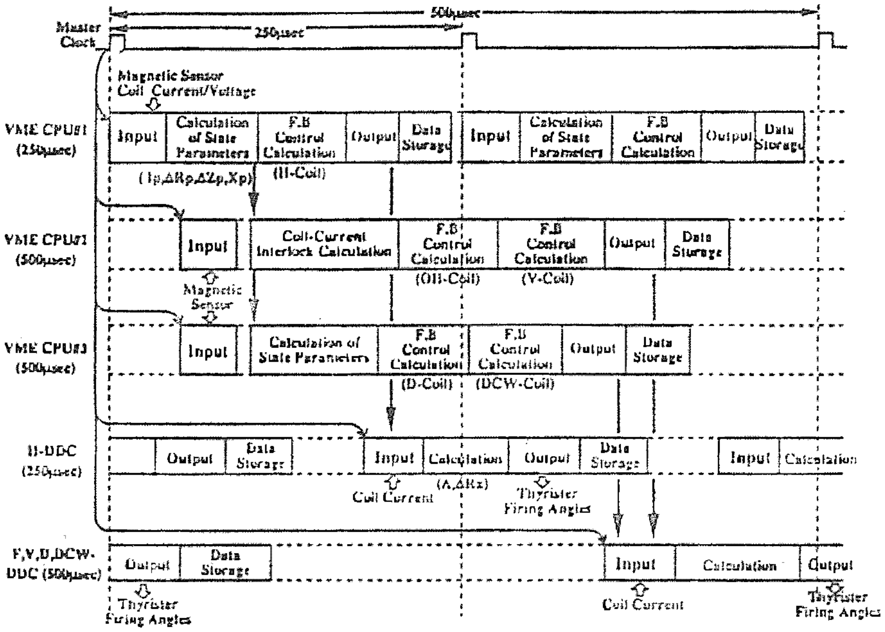


Fig. 3 Control time chart of the VME multi-processor system

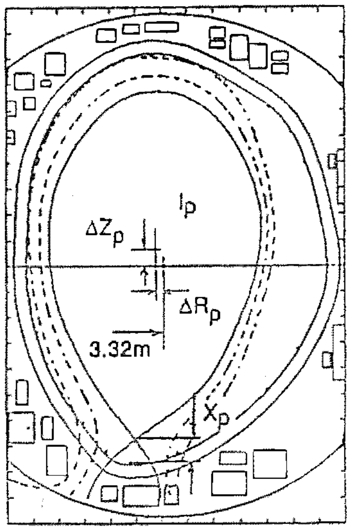


Fig. 4 JT-60U plasma shape and the controlled parameters

3.2 *Plasma Control Performance*

Figure 4 shows the shape of the JT-60U plasma along with its controlled parameters, i.e. the plasma current (Ip), the plasma horizontal position (ΔRp), the plasma vertical position (ΔZp), and the diverter X-point position (Xp). In the initial operation of the JT-60U, the feedback control system well controls these parameters by PD (proportional and differential) control with matrix gain and stable divertor plasmas with plasma current of 5MA have been obtained. As an example of the control performance, the step response of the vertical position ΔZp is shown in Figure 5. The amplitude of the fluctuation is less than 5 mm and the settling time is less than 10 msec short, though we observe the overshoot which may not give bad influence on plasmas.

4. SUMMARY

The VME multi-processor system, where three RISC computers are adopted, have been newly developed for the JT-60 plasma control. The new system makes it possible to execute more accurate and faster control of the plasma position and shape. In the initial JT-60U experiments performed from the last April through October, stable divertor plasmas with the current of 5 MA have been obtained with this control system.

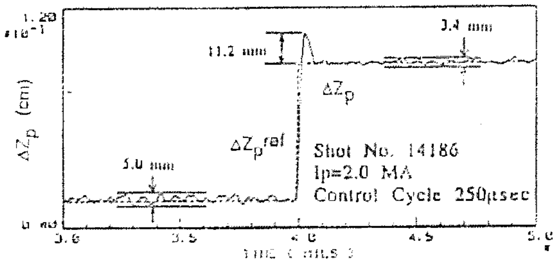


Fig. 5 Step Response of the Plasma Vertical Position in the Feedback Control

ACKNOWLEDGEMENTS

The authors wish to thank the other members of Division of JT-60 Facility I at JAERI for their collaboration in the development of this plasma control system.

REFERENCES

[1] T. Kimura, et al., "VME Multi-processor System for Plasma Control at the JT-60 Upgrade," IEEE Trans. on Nuclear Science, vol.35, 1989, pp.1554-1558
[2] M. Matukawa, et sl., "JT-60U System Design," in Proceedings of the 15th Symposium on Fusion Technology, Utrecht, 1988.

Very Fast Feedback Control of Coil-Current in JT-60 Tokamak

T. Aoyagi, T.Terakado, M.Takahashi, H.Nobusaka, J.Yagyu and Y.Matsuzaki
Japan Atomic Energy Research Institute, Naka Fusion Research Establishment,
Naka-machi, Ibaraki-ken, 311-01, Japan

Abstract

A direct digital control (DDC) system is adopted for controlling thyristor converters of power supplies in the JT-60 tokamak built in 1984. Microcomputers of the DDC were 5 MHz i8086 microprocessor and programs were written by assembler language and the processing time was under 1ms. They were, however, too old in hardware and too complicated in software. New DDC system has been made in the JT-60 Upgrade (JT-60U) to control the power supplies more quickly under 0.25 and 0.5 ms of the processing time and also to write the programs used by high-level language. The new system consists of a host computer and five microcomputers with microprocessor on VMEbus system. The host computer AS3260 performs on-line processing such as setting the DDC under the discharge conditions and so on. Functions of the microcomputers with a 32-bit, 20 MHz microprocessor MC 68030, whose OS are VxWorks and programs are written by C language, are real-time processing such as taking in instructions from a ZENKEI computer and in feedback control of currents and voltages of coils every 0.25 and 0.5 ms. The system is now operating very smoothly.

I. INTRODUCTION

Control of a current, positions and configurations of plasmas in a tokamak is done by poloidal magnetic fields, and power supplies of the poloidal field coils have to be controlled very fast to suppress intrinsic instabilities of plasmas. A schematic diagram of the feedback control system is shown in Fig.1. Magnetic probes measure magnetic fluxes of plasma and a ZENKEI real-time control computer[1] performs as follows: calculating the positions of plasmas and the derivations of the reference positions, multiplying PID gains to the derivations and outputing the command of the coil-currents. Direct digital control (DDC) of JT-60 poloidal field power supply (PFPS)[2] carries out that taking in commands of the ZENKEI ($I_F{}^{ref}$ in Fig.1) and giving out the delay angle cosine (Ec) to phase controllers (PHC) of thyristors. Thyristor banks have two sets of the converters which deliver the plus and minus direction currents (I_1 and I_2), respectively, and during low-current under 20% of the rating coil-current two converters supply circulating currents (I_c) to operate the thyristors smoothly.

II. DDC SYSTEM

A. *Functions*

The DDC performs two functions in details such as on-line processing during no-discharge and real-time processing during discharge. Functions of on-line processing are as follows: (1) setting the DDC system under conditions of the discharge instructed by the ZENKEI, which are in detail instructions and checks on the discharge mode of CAMAC modules and of PHC and are diagnosis of them before the

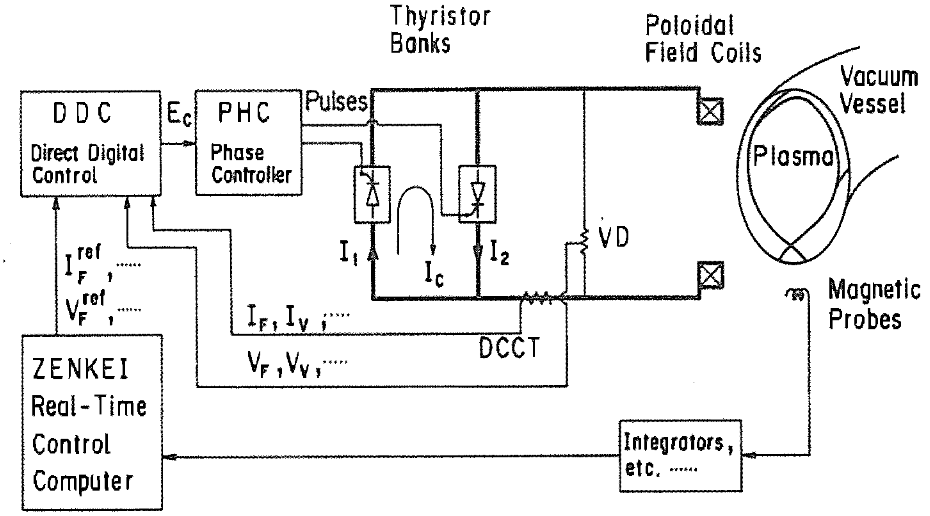


Figure 1. Schematic diagram of feedback control system of plasma position

discharge, (2) managing the time-sequence and (3) collecting and transferring data of coils' currents and voltages to the ZENKEI after the discharge. Functions of real-time processing are taking in instructions of a feedback control computer IIb of the ZENKEI and in currents and voltages of coils which are measured by direct current current transformers (DCCT) and voltage dividers (VD), calculating minimum time response control and non-interactive control algorithms and giving out the Ec value to the PHC every 0.25, 0.5 and 1 ms as shown in Fig.1.

B. Original and old DDC system

The original DDC system consists of five CAMAC crates correspond to five converters of PFPS and one crate for collecting data. Each crate has many modules as shown in Fig. 2. Functions of the modules are as follows. An ACM module with a 5 MHz i8086 microprocessor is most important module and performs almost all control of the DDC. An ACB module is communication crate between the ZENKEI and the DDC CAMAC by branch highway (BH). An INTR module is taking in the timing clock and the interacting events. Many AI modules are taking in coil-currents and voltages, DO modules are giving the instructions of start/stop of operation, on/off of bypass pair(BPP) operation[2] and Ec values calculated to PHC.

Purposes of adopting the DDC in PFPS are not only fast processing, but also having the flexibility in the control. However, the programmings written by assembler language were very complicated and were very hard to change. Because the 16-bit microprocessor was very primitive in 1984 and it had to execute many processing during 1ms. In order to solve the problems, a developing a new system was tryed. In the system a mini-computer ECLIPSE-MV was used instead of the ACM and Ada was used as the language. The system could operate preliminarily, but it was special and expensive.

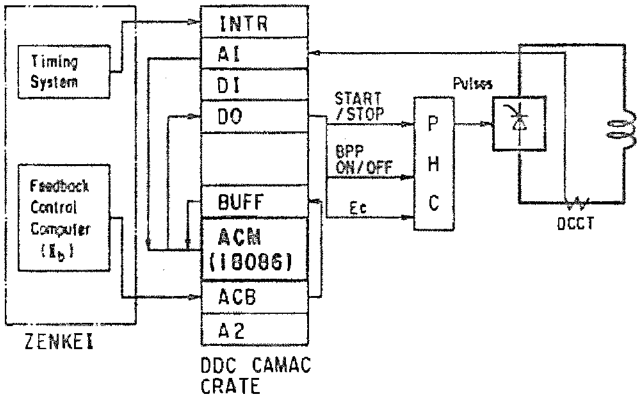


Figure 2. Schematic diagram of original CAMAC carte of DDC system in PFPS

III. NEW DDC SYSTEM

A. Objectives and configurations of new system

In JT-60U, plasma current increases to 6 MA and plasmas have large volume of 100 m^3 with vertically elongation[3]. In order to suppress intrinsic vertical instabilities of elongated plasmas, there needs faster processing and more flexible control in DDC to change the control algorithms easily and to protect over-currents of coil-power-supply systems[4]. The target of processing time is under 0.5 ms and is hopeful under 0.25 ms[5]. The flexibility of the control is as follows; (1) switching operation modes from circulating current mode to non circulating current mode, (2) changing control method from coils' current feedback control (ACR) to coils' voltage control (VR), (3) starting and stopping thyristor converters with many patterns such as gate-shifting, BPP and so on. For these purposes old DDC microcomputers and their softwares were too old in hardware and too complicated in software.

In order to fulfill above-mentioned requirements, a new DDC system had to be made, and the configurations of the system are as follows. First, it has been adopted to a newest and fastest microcomputer to write programs used by high-level language C and to execute the floating calculation, because the programmings and maintenance of the system will be easy. Second, the system has been made under the environment of UNIX, because the UNIX has been ready for debugging, editing, using many application programs and so on very strongly. Third, the network had to be adopted to the standard communication software, because there need no programs of communication. Finally, we divided the functions possessed original microcomputer i8086 in the on-line processing and the real-time processing. Because we want to program under the environment of useful and strong operational system(OS) which is equipped with a work station(WS). The WS, however, is very hard to process the real-time processing very fast, so this processing is executed by the microcomputer under an adequate OS.

B. Hardwares

An architecture of DDC system developed and made for the JT-60U is shown in Fig. 3. The main components of the system are a host computer AS3260 (Toshiba Corp.) , five microcomputers MVME 147 (Motorola Inc.) and five CAMAC crates correspond to five converters of the PFPS. The AS3260 which is the same specifications of WS Sun 3/60 is connected with a ethernet cable of the ZENKEI by a 8-channel transceiver and a bridge, and also connected with a ethernet cable of MVE crates by two 8-channel transceivers, two optical remote repeaters and two splice boxes through a optical fiber. The microcomputers which consist of a 32-bit, 20 MHz microprocessor MC 68030 and a floating point coprocessor MC 68882 on the MVEbus are mounted on chassis possessed VME backplans and power supply. The chassis encloses a CAMAC branch drivers CBD 8210

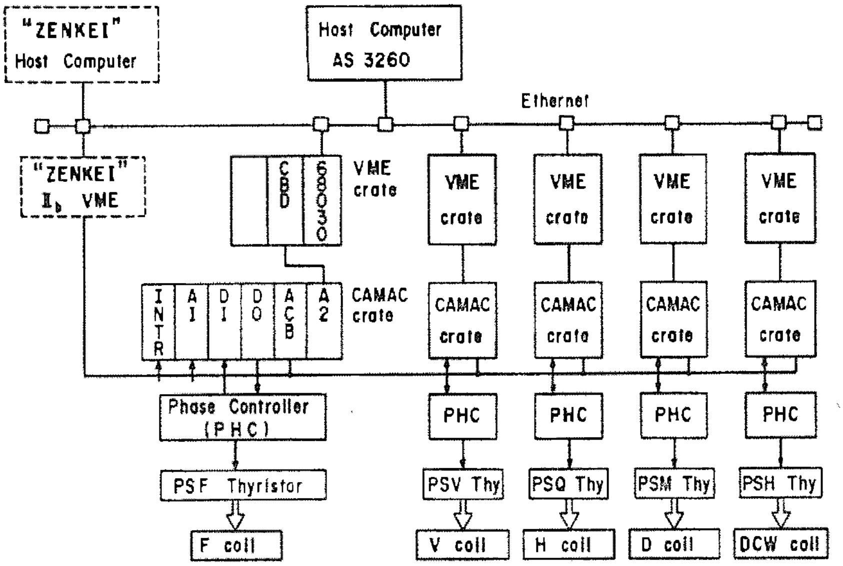


Figure 3. Architecture of new DDC system of poloidal field coil power supply

(Creative Electronic Systems) which connects the VME crate and a A2 module of the CAMAC crate by the branch highway and a I/O transition module MVME 712M. Except the ACM module, the CAMAC crate is quite same as the original architecture explained as the former section, namely it consists of a ACB module which is connected the ZENKEI IIb by BH and DO, DI, AI and INTR modules which are connected a PHC, DCCTs and so on. The system has still another hardwares such a VME crate for collecting data and a WS AS3160 and a CAMAC crate for testing the system

C. Softwares

An organization of the DDC software is shown in Fig.4. The host computer whose OS is UNIX performs on-line processing described in section II.A. The main programs are an on-line processing task, a ZENKEI communication task and a microcomputer communication task. The configuration of the ZENKEI communication task are as follows: (1) the processing execute each power supply separately, (2) the communication protocol is TCP/IP, (3) the connection between the ZENKEI and the AS3260 is set by actively and (4) the client is the ZENKEI IIb and the server is the AS3260. Second, the configuration of the microcomputer communication task are nearly the same as that of the ZENKEI communication task only except that the client is AS3260 and the server is each microcomputer. Finally, the on-line processing task performs as follows: (1) setting the DDC system under conditions of the discharge such as reasonable check of the power supply circuit and of constants values of many control tables and diagnosis of the crates and PHC before the discharge instructed by the ZENKEI and then answering back, (2) collecting and trans-ferring data of coils' currents and voltages to the ZENKEI.

Microcomputers whose OS are VxWorks (Wind River Systems, Inc.) carry out real-time processing. The main programs written by C language are a real-time processing task, an interrupt processing task and a communication task. The communication task performs watching and executing of the command to the CAMAC and the referencing/changing memories of CPU instructed from the host computer by TCP/IP protocol. The real-time processing task is set by the main task of the CPU, and then it begin to execute the real-time processing at the time of LAM (look at me) interrupting signal from CBD. The sequence of the real-time processing is as follows: it catches the LAM signal from the ADC of the CAMAC, then it takes in currents and voltages of coils,

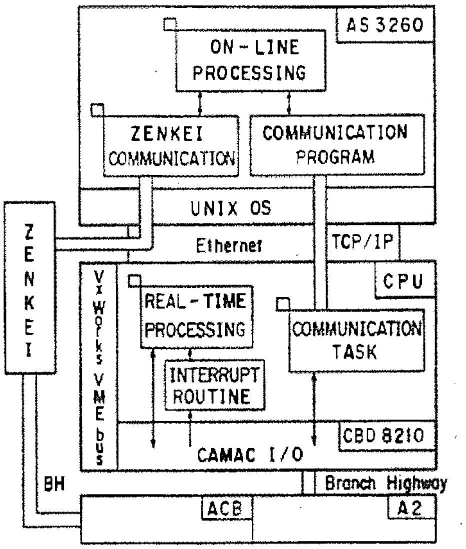


Figure 4. Organization of new DDC software

calculates the control-algorithms and gives out the Ec value to the PHC every 0.25, 0.5 and 1 ms. Finally, the interrupt routine works from the interrupting signal of the CBD. The routine performs resets of the PHC by the BPP-command and of the Ec values of the PHC at the repair of thyristors, and it do also start-processing of the real-time processing.

IV. TESTS AND RESULTS

New DDC system has been made and tested. A processing time which is measured from input of data to output of Ec calculated is shown in Table 1.

	PSF	PSV	PSH	PSQ	PSM
ACR MODE					
With C.C	328	350	278	299	——
Without C.C	258	264	230	235	185
VR MODE					
With C.C	304	335	261	294	——
Without C.C	267	267	237	238	190

ACR: Auto Current Regulation
VR : Voltage Regulation
C.C : Circulating Current

Table 1. Processing time of each power supply (μs)

Results are satisfied by achieving the target that the processing time is under 0.5 ms. Moreover, we achieved the processing time of 216 μs in PSQ with C.C mode by changing the clock time of the microcomputer MC 68030 from 20 MHz to 32 MHz [5].

Figure 5 shows waveforms of currents on working of PSH power supply. An upper waveform is a current of plus direction, a middle one is a minus one and a lower one is a coil current summed plus and minus currents. At a start phase, PSH1 and PSH2 power supply which deliver plus and minus current, respectively work together. At the current of 11 kA which is a half of the rating current, PSH1 stops and only PSH2 works. The constant current of 12 kA is continued during 1.5 second, and then it decreases. At a point of 4.4 kA which is 20% of the rating current, PSH1 works again, and then the coil current switches from minus to plus. At the end of the operation, PSH1 and PSH2 are to terminate in gate-shifting operation[2], namely in inverter operation for reducing the coil current quickly, and the results are very satisfactory. In order to achieve this operation, some new algorithms and computer simulations were performed.

The new DDC system has been made, tested and operated on five coil-currents' power supplies, and the results showed that the system operated very smoothly about the very sophisticated control such as switching operation modes, changing control method from ACR to VR and starting and stopping of BPP of thyristor converters during the real-time control. Moreover, the new system was applied to controlling the vertical instability of plasmas, and the results showed that the oscillation amplitude of the vertical instabilities decrease from 4.5 cm to 3.2 cm as changing the clock time from 1 ms to 0.25 ms, respectively [5].

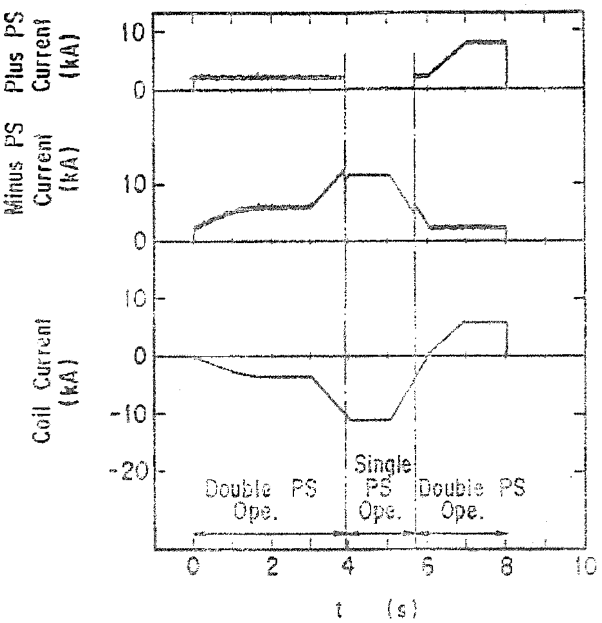


Figure 5.
Waveforms of currents controlled by new DDC system

V. CONCLUSIONS

We have modified direct digital control (DDC) system of thyristors converters in JT-60 poloidal power supplies to process quickly and to have the flexibility in the control. Results of operations showed that these modifications have been performed very good.

ACKNOWLEDGEMENTS

We would like to thank Mr. K. Ujiie for his helpful discussions as DDC. We also wish to thank our colleagues of JT-60 Power Supply Group for help with these works.

REFERENCES

1. I. Kondo et al., Fusion Engineering and design 5 (1987) 69
2. R.Shimada et al., ibid. 47
3. H.Ninomiya et al., Plasma Devices and Operations, 1(1990) 43
4. Y. Matsuzaki et al., Proc. of 16th SOFT (1990) 1482
5. M.Takahashi et al., this conference

Workstations as Consoles for the CERN-PS Complex, setting-up the environment.

P. Antonsanti, M. Arruat, J.M. Bouche, L. Cons, Y. Deloose, F. Di Maio.
CERN PS - 1211 Geneva 23 - CH

Abstract

Within the framework of the rejuvenation project of the CERN control systems, commercial workstations have to replace existing home-designed operator consoles. RISC-based workstations with UNIX® , X-window™ and OSF/Motif™ have been introduced for the control of the PS complex. The first versions of general functionalities like synoptic display, program selection and control panels have been implemented and the first large scale application has been realized. This paper describes the different components of the workstation environment for the implementation of the applications. The focus is on the set of tools which have been used, developed or integrated, and on how we plan to make them evolve.

I. INTRODUCTION

The current control system of the PS complex is based on 16 bit computers which will be replaced because of obsolescence of the hardware and system software. A rejuvenation project is in progress for upgrading the different parts of the control system: hardware interfaces, process computers, communications and operator consoles [1] [2]. UNIX has been selected for both the console layer and the process layer.

During this first 3-year period (1989-1991), a workstation infrastructure has been set up and the basic building blocks of the programming environment have been provided. The first large scale application, the hadron injection process into the PS, has been realized for the 1991 PS complex start-up and extended during this time. This paper describes the infrastructure and the programming environment. The application programs and the user interface are both described in separate papers [3] [4]. In the context of such an evolution, the first task is to compose a base environment whose major parts are: hardware, system software, data-base management system, equipment-interface and user-interface. From experience, we were very concerned about getting as many functionalities as possible from this layer in a "safe" way: we wanted to minimize system development and be confident in the future of the environment.

The second task is to provide generic applications to support functions like console management, error handling and all direct interface with the equipment: synoptics of parts of the machine, parameter tables and control panels. This has been achieved through collaboration between the controls group and the operation group.

The third task is to integrate into the environment additional user-oriented tools for the production of specific applications and for simplification of generic tools. These tools are mostly from the commercial market and therefore it is certainly the fastest changing part.

II. BASIC ENVIRONMENT

A. Hardware infrastructure

For operation and for development, DEC™'s RISC-Ultrix™ workstations are used (about 50 in 91/92). We use common configurations with only network interface (i.e. no direct VME, CAMAC or GPIB) and local disks for virtual memory and temporary files only.

The central facilities consist of servers providing the following services: workstation system files, user files, data-base and time-sharing servers. Central time sharing servers are used mainly for resource hungry software (hardware or administration) which are transparently available on office workstations by means of the X-window network facilities.

Each local sub-network includes a regional server supporting local workstations, DSC[1]s and data. These servers are high-end workstations with SCSI disks.

One important characteristic of our current architecture is that in order to cope with man-power resources for exploitation, we opted for a very homogeneous environment. Every operation critical system is, for the time being, from a single vendor and covered by a single maintenance contract (hardware and software). This has been very efficient. However, for the sake of real vendor-independence, software portability and commercial relations, mixing vendors would be profitable, especially for tasks which are not exploitation-critical.

Another characteristic of this architecture is that our newest servers are enhanced workstation configurations instead of "mid-range" systems with high performance bus, fast dual-ported disks, etc. This is due to the increasingly faster obsolescence of the hardware and the fact that Ultrix does not

---

® UNIX is a registered trademark of UNIX System Laboratories.
™ X Windows is a trademark of Massachusetts Institute of Technology.
™ Motif is a trademark of the Open Software Foundation.

™ DEC and Ultrix are trademarks of Digital Equipment Corporation

[1] DSC are VME-based process computers with Real-Time UNIX.

provide yet full benefit of DEC's special hardware such as BI and DSSI.

B. System Software

The base system software is the common Ultrix distribution: UNIX programming environment, TCP/IP, NFS, etc. with minimal additions from public-domain or commercial sources. To remain vendor-independent, we try to follow the evolution of industry standards as close as possible, especially OSF[1]'s products. We have replaced DEC-windows™ with Motif since 1989. We did not anticipate the distribution of the OSF/1™ operating system nor of DCE™ (Distributed Computing Environment), but we will use the vendor's versions.

The basic programming environment on our platforms consists of a C programming environment, composed of the UNIX tools (compiler, debugger, Make, SCCS...) with the addition of GNU Emacs[2] and of interactive tools, like DEC-Fuse™ .

C. Data base

Historically, the first major additional component to the base system software is a data management system. We are using a commercial relational data-base: Oracle™ which is the standard database in use at CERN. Most of the control system data is managed through it: equipment definition, alarm codes or computer definitions, for instance.

A client interface is available on every system. Forms and SQL are used for the various software exploitation tasks: software module additions, equipment updates, etc. Programs manipulating complex or archivable data, like the beam's cycles editor and the error servers use a central data-base via embedded SQL.

A dedicated server is providing "on-line" Oracle service for the two accelerator divisions. In addition, critical and read-only data which require efficient access, like equipment description, are distributed from Oracle into dbm[3] data-bases on the different sub-networks.

D. Equipment Interface

The next building block is communication with the hardware. An equipment access interface is available on process computers: old ones (Norsk Data® and PDP11®) and new ones (DSC). Remote procedure calls (RPC) are used to communicate with the process computers from the workstations.

The equipment interface in the CERN-PS complex is a well-defined, strict syntax interface. The arguments of the main procedures are: equipment identifier, "property" (selector inside a fixed set), event condition (CERN-PS timing) and data. The data is constrained to simple types (char, int, double, etc.) and to one dimension only. There is also a very limited set of procedures.

All the equipment definition (name, class, host computer, number, description, etc.) is maintained in a central data-base.

These two aspects have many advantages for integrating equipment access into any environment: small number of procedures and simple argument types make it easy, even in a commercial tool like a spreadsheet. In addition, once all the different communication channels are supported, any equipment defined in the data-base is accessible, without explicit knowledge of the network topology.

The equipment interface in the workstations is a local dispatcher to the equipment interface in the process computers. It provides network abstraction: the equipment identifiers have no correlation with their physical implementation (network, host...). This is mandatory in order to be able to move equipment from old computers to new one without interfering much with the applications.

An additional software layer is provided for generic applications which need to operate on a wide range of equipment types without integrating equipment-specific code. For example, how to display the status of a power supply or how to control it, need not to be re-defined in each generic application. All specific data and code related to the different equipment class, like labels, control words or transition functions are maintained in this library layer.

E. User Interface

The last major part of base environment is the user-interface.

This part is mainly composed of the Motif tools: interaction objects ("widgets") like buttons, menus or dialog windows, general libraries and a user-interface definition language (UIL). Motif is an additional layer on top of X-window and we try to ensure that programmers need only use this upper layer (i.e. minimal use of X-lib).

The basic widget set in the Motif distribution address most of the user-interaction issues. However, it does not provide many tools for data presentation nor for complex parameter control. These two parts requires programming at a lower levels: X-window (Xlib) or toolkits "intrinsics".

One solution in avoiding the duplication of such functions by different programmers is to add widgets for unsupported functions. Adding library functions like graphs in the form of additional widgets has the benefit of providing a uniform final programming environment. As no widgets were available as a common well-accepted solution, we developed our own widgets for these functions.

---

[1] OSF : Open Software Foundation

™ DEC windows is a trademark of Digital Equipment Corporation

™ OSF/1 and DCE are trademarks of the Open Software Foundation.

[2] GNU Emacs is a programming oriented editor from the Free Software Foundation.

™ DEC Fuse is a trademark of Digital Equipment Corporation

[3] dbm is a simple data-base system in UNIX

We provide programmers with widgets for data presentation: alpha-numeric or graphics which also support some major features in our environment like high refresh rate without blinking or variable parameter types.

We also provide a single widget for parameter control with dedicated functionalities like familiar interface (compared to previous hardware devices) or multiple input modes (mouse, arrow keys, numeric key-pad..).

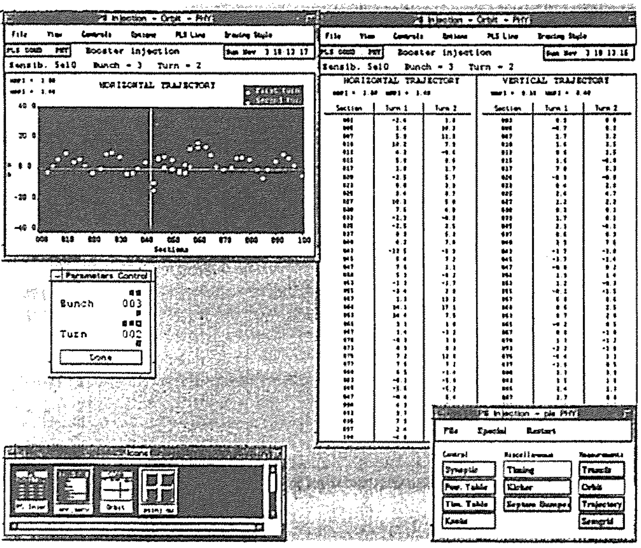


Figure 1. Motif application using home-made widgets

The development of widgets is a rather costly process which requires some proficient programmers but developing these functions as widgets has simplified greatly most of our Motif applications.

For end-users, the Motif programming environment is a very complete environment but requires some specific training. Courses have been organized and a support activity has been provided.

We are now introducing interactive editors for producing user-interface definition in a Motif-standard format: UIL or C. Such tools are available from various sources, like DEC or Siemens. If strictly based on Motif, without any specific data-structure, library or language, they have limited prototyping facilities but produce tool-independent applications.The major work in integrating these tools in our environment is to include our "user-defined" widgets which is an extra cost to their development.

III. GENERIC APPLICATIONS

Generic applications are programs which are not bound to a specific operation nor to a specific piece of equipment. The two main categories are general console utilities and equipment oriented applications.

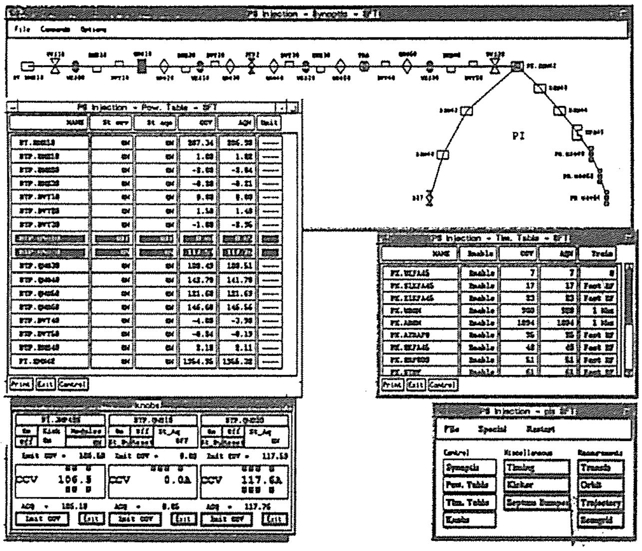


Figure 2. Generic applications

A. Application management

A console utility is required for handling the different functions available when the user logs in: program presentation, global condition selection and general facilities control (alarms and errors).

Some window management is also required in addition to the functions which are already available via the Motif window manager. The initial position and size of the windows need to be managed in order to distribute the application windows on the screen properly. The windows of the different programs which belong to the same operation context (same accelerator, same condition) also need to be handled as a set: common identifiers in the title bar and global iconification/raising/destruction. In addition, starting an application, bringing its window to front or "de-iconifying" it are connected to a unique interaction element.

One major issue in this area is window overlapping. In our context, the set of concurrent application which an operator will activate is not well defined; it is then difficult to define a fixed layout. Therefore, screen zones are defined mainly through program types. For example: control panels should appear in the bottom left corner of the screen, while equipment tables are on the right part.

B. Parameter tables

The first equipment-oriented generic application is a dedicated application for displaying operational parameters in a tabular format.

Each piece of equipment (power-supply, timing, etc.) belonging to a user-defined "working set" is presented along with a list of equipment-related values like status, control/acquisition value, unit and a color indication. The tool can be set to refresh the data every machine cycle if necessary (1.2 s).

The functions of this tool are multiple: to present the list of equipment, to support the selection by the user of a particular piece of equipment, to provide an interface for simple commands (On, Off ...) and to activate control panels on request.

C. Synoptics

Another implementation of the same kind of functionalities are synoptics presentations of a part of the machine.

Synoptics present the major and critical control element of that part of the machine and indicate their status by means of the color. They also present the position of some beam monitors. Like parameter tables, they include selection, simple commands and control panel activation.

These displays provide less detail than tables (no current values) but in a more synthetic way. They are used mainly for beam-transfer sections and not yet for circular parts.

The major problem for their implementation is that synoptics require user-interface information, which is not part of the equipment description available in the control system data-base. In our context, two types of information were not registered in data-base: the topological information which is used for positioning the icons and the type of physical element which is necessary for selecting the icon to display.

We opted for a data-driven solution : topological information and equipment specifications are entered in a synoptic-specific file. Another synoptic application realized for the control of the Proton Linac [5] includes a graphic interactive editor for entering this information.

Interactive editing, based on home-made tools or on commercial ones (data-presentation packages or user-interface editors), provides the end-user with a complete solution for building the synoptics themselves. Otherwise, one more maintainable solution is to exploit central data-base information shared with other programs, like AutoCAD™ or MAD but such a database does not exist yet in our context.

D. Control panels

A third important generic application is the direct interaction with pieces of equipment by means of "control panels", which are dedicated dialog boxes.

The user can activate individual control for a particular piece of equipment (power-supply, kicker, gun...) from different parts of the environment: from the application management layer, via buttons, from the parameter tables and the synoptics, via double clicks on the icon or via menus. Individual control activation can also be implemented in user-tools, like spreadsheets by means of an external function.

Wherever the request is made from, the control panels are handled by the same server application. It was implemented in this way for the sake of homogeneity and interface simplicity. As an additional homogeneity feature, all the control panels activated from the same operation context appear in the same main window (i.e. "shell window"). The exact layout of the control panels depends on the type of the equipment; however, parameters are usually presented in a similar way : name, buttons for discrete commands, display of the status, a "wheelswitch" widget for controlling the continuous parameter, the display of the parameter acquisition, etc.

We are modifying the tool in order to be completely data-driven : we do not want to have to extend the tool for each new type of equipment. Therefore, all the data and code which are necessary for the user-interface are moved to the equipment-interface library.

IV. USER-ORIENTED TOOLS

The basic Motif environment is very suitable for producing interactive applications. However, end-users as well as generic application makers can greatly benefit from the integration of higher-level or more user-friendly tools.

Most of these tools are from commercial sources and, as Motif is a rather recent market, the majority of them are just starting to be available in production version.

The integration of a commercial tool usually requires an interface to our specific functions, like equipment-interface and to provide support to end-users if possible.

A. Nodal and Console Emulation

The first user-oriented tool is CERN-sourced. Nodal is an interpreter language which is widely used in control applications at CERN.

The workstation version includes the equipment interface and some user-interface facilities based on Motif [6].

In addition, most of the facilities which were supported on the previous consoles (touch-panels, alpha-numeric displays, knobs) are available by means of emulation through compatible functions.

Nodal was the major programming environment for the consoles and is very familiar to CERN's staff. The Nodal environment on the workstations provides a very valuable way of porting existing applications and introduce many users to the workstations.

Having emulation facilities is critical in order to be able to transport, in a short time period, the whole set of applications of one accelerator from traditional consoles to workstations while focusing only on a few critical parts of them. It also adds some familiarity in the end users context.

B. Data Presentation

In the X-window/Motif environment is now possible to acquire commercial tools, like DataViews™, which includes a lot of data-presentation functions, like multiple 2D and 3D graph formats. Some also include a Motif look-and-feel (buttons shadows...) which provide consistent presentation and

---

™ AutoCAD is a trademark of Autodesk Corporation.

™ DataViews is a trademark of V. I. Corporation

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI01

interaction for end-users. Such tools usually support only a sub-set of the Motif facilities for handling the user interaction but are very powerful for data-presentation.

Such a tool can be very useful for the implementation of synoptic-like applications, i.e. applications with dynamic colors, shapes or graphs but without much computation or many options. This can provide the end users with a very productive solution for synoptic-like applications.

We also plan to integrate PHIGS as soon as there is a need of a high-level graphics library.

C. Motif-based

Another major category of tools are user-interface building tools with dynamic programming support such as interpreted languages. We'll provide such a tool (UIM/X™) for fast-production of simple applications.

The environment is a graphical editor for building up the user-interface of the application and a C interpreter for testing the application from the editor. By-passing most of the UNIX compilation process, the development of the application is much improved. Programmers still need to be trained to Motif but can use a very efficient environment.

One limit of such solutions is that applications are tool-dependent instead of being just Motif-dependent. However, run-time support is usually not a big financial issue and it can be envisaged to re-work the application when it is stable enough in order to transform it to Motif-only (some tools include this facility).

D. Spreadsheet

High-level spread-sheet programs, like Microsoft-Excel™ are every-day tools of much of the scientific staff. These tools include many presentation facilities and embedded functions. However, up to a recent date, the Motif market was rather poor in this area. As products are appearing now, we are integrating them (we have started to do this with Wingz™)

These tools have many applications: quick prototyping of control algorithms by the specialists themselves, dynamic treatment and presentation of the data, etc. One major area for them are the machine development applications. The control of the CERN Isolde separators is strongly based on such tools and this application of a spreadsheet is very spectacular [7].

We are porting one off-line PC application to on-line. As one production version of this program has been already realized in the plain C-Motif environment, we will be able to compare the two solutions in area like flexibility, development and maintenance cost, user-interface efficiency or robustness.

E. Mathematical package

Mathematical packages including symbolic computation and graphical data-presentation are becoming popular tools among our scientific users. Mathematica™ is a good example and we are providing this tool on workstations.

Such a package can be used for prototyping algorithms in a very efficient way provided that the specialist is used to it or is assisted by an expert in the package.

One interesting feature of these tools is that a production version of the program may be implemented in an operation-oriented environment supporting standard facilities (instruments control, timing conditions, archives...) with communications to a separate mathematical process supporting the computation.

Fortran is also available and the MAD optic program is currently being ported.

V. CONCLUSION

One main objective of this first period dedicated to set-up the workstation environment was to provide a complete production environment with low costs in development and maintenance of the system software and with small man-power resources for the system exploitation. The current situation is rather satisfying from this point of view. The environment has proved to be convenient for application production, although there are still a lot of improvements to introduce, both in performances and functionalities.

The two more important events which occured during this set-up phase are the suddenly increasing range of solid industrial standards supported by the major hardware vendors and the jump of the software vendors into the Motif market.

There are probably two directions where we will improve our environment and expertise. The first one is the integration of analysis and design tools to enhance production and maintenance of complex "made to measure" programs. The second one is to continue the effort of implementing user-oriented tools for making life easier for application-makers or to enhance (replace?) the specification process.

V. REFERENCES

[1] The PS and SL Control Groups, PS/SL Controls Consolidation Project, Technical Report, CERN PS/91-09 (CO), CERN SL/91-12 (CO), April 1991.

[2] R. Rausch, Ch. Serre (Editors), Common Control System for the CERN PS and SPS accelerators, Nov. 1991, these Proceedings (ICALEPCS, Tsukuba, Japan, Nov. 1991).

[3] M. Arruat, M. Boutheon, L. Cons, Y. Deloose, F. Di Maio, D. Gueugnon, R. Hoh, M. Martini, K. Priestnall, J.P. Riunaud, "New Controls for the CERN/PS Hadron Injection Process using Operating Tools and High-Level Modelling Accelerator Programs", these Proceedings (ICALEPCS, Tsukuba, Japan, Nov. 1991).

™ UIM/X is a trademark of Visual Edge Software Ltd

™ Excel is a trademark of Microsoft Corporation.

™ Wingz is a trademark of Informix Software, Inc.

™ Mathematica is a trademark of Wolfram Research, Inc.

[4] M. Boutheon, F. Di Maio, A. Pace, General Man-Machine Interface used in accelerator Controls: some applications in CERN/PS Control system rejuvenation, Nov. 1991, these Proceedings, (ICALEPCS, Tsukuba, Japan, 1991).

[5] J.M. Nonglaton, U. Raich, Porting Linac Application Programs to a Windowing Environment, Nov. 1991, these Proceedings, (ICALEPCS, Tsukuba, Japan, 1991).

[6] F. Perriollat, G. Cuisinier, A. Gagnaire, "Nodal", PS/CO Note 91-20, November 1991.

[7] R. Billinge, A. Bret, I. Deloose, A. Pace and G. Shering, "A PC Based Control System for the CERN Isolde Separators", these Proceedings (ICALEPCS, Tsukuba, Japan, Nov. 1991).

General Man-Machine Interface used in Accelerators Controls: Some Applications in CERN-PS Control Systems Rejuvenation

M.Boutheon, F.Di Maio, A.Pace
PS Division
CERN CH 1211 Geneva 23 Switzerland

Abstract

A large community is now using Workstations as Accelerators Computer Controls Interface, through the concepts of windows - menus - synoptics - icons. Some standards were established for the CERN-PS control systems rejuvenation. The Booster-to-PS transfer and injection process is now entirely operated with these tools. This application constitutes a global environment providing the users with the controls, analysis, visualization of a part of an accelerator. Individual commands, measurements, and specialized programs including complex treatments are available in a homogeneous frame. Some months of experience in current operation have shown that this model can be extended to the whole project.

I INTRODUCTION

When the decision was taken to rejuvenate the computer control system operating the CERN-PS accelerators complex [1], it was felt that users should define their needs [2]. More precisely, the end users, i.e. the operation teams had to give their views on interaction principles and tools.

The framework of this study was of course delimited by the now worldwide accepted notion of G.U.I.[1] , integrating the concepts of windows, pull-down menus, pop-up menus, icons and objects selection, all these being driven by a powerful multitasking system [3]. Taking into account the dimension of the process - the PS complex includes 10 accelerators - a prototype had to be constructed in order to evaluate the new human interface proposed.

The hadron beam transfer line from the 1 GeV Booster synchrotron and the related CPS injection process were selected as guinea pigs. The principles and applications are described below.

II PROCESS STRUCTURING

In a very large process to be controlled from a centralized point, the first task consists in defining a structure allowing each member of an operating team to work in a quasi independent and secure way. These principles were already successfully introduced in the present control system [4] and are kept here. Moreover, the PS accelerators complex pulse-to-pulse modulation (PPM) working mode [5] imposes the notion of virtual machine: a parallel adjustment of concurrent beam types in the same accelerator is possible.

From the above the concept of an Application emerged: the whole lot of application programs needed to operate a logical part of an accelerator in an autonomous manner. We are talking here of the "CPS 1 GeV Injection Application" given as an example.

An Application includes:

- the complete access to the control/acquisition of the parameters set composing the sub process
- the controls of the dedicated measurement devices and associated presentation programs
- the temporary specialization of some general measurement devices (dedicated initialization of parameters)
- access to particular application programs developed for the specific sub process.

The term Application will be used in what follows to designate the working environment defined above.

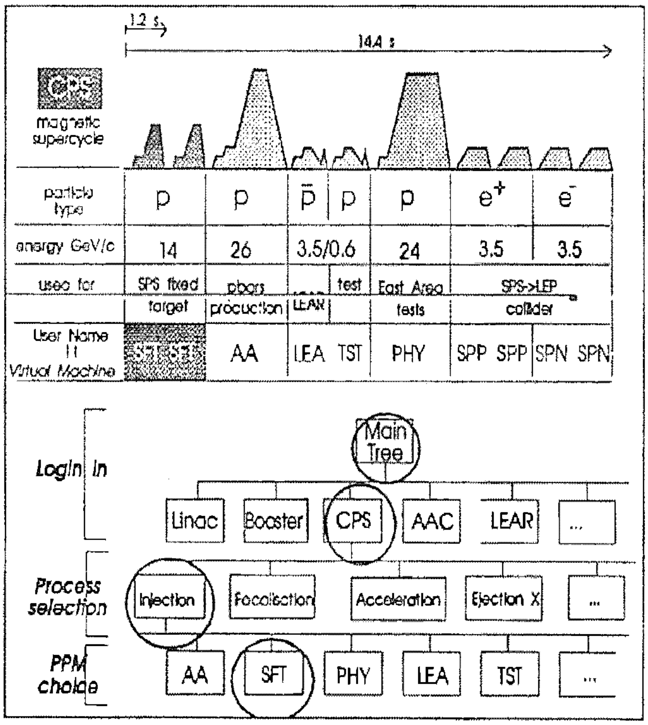


figure 1: a CPS supercycle showing the succession of beam types and selection structure scheme. Illustrated: The 1 GeV Injection Application addresses to the CPS parameters valid for the beam sent to the SPS accelerator.

---

[1] Graphical User Interface

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI02

The complete design of the user interface for the computer control system will consist of specifying all the Applications. The structure is Beam Oriented rather than Hardware Oriented as suggested by figure 1.

III TYPE OF CONTROL TOOLS

We considered in this study that the only interacting tools would be a mouse, and a keyboard. The (software) tools contained in an Application can be classified according to the different stages of a process. We have selected:

Synoptics

They are used to visualize the whole process covered in the Application, namely the pictorial display showing the pieces of hardware constituting the sub processes, on which the user can act. This figure gives on line the status of the apparatus, at least along the beam path if exact geographical representation is not possible. Moreover the synoptics is made of selectable objects through which other tools can be accessed.

Individual Controls

Each piece of hardware in the Application must be accessed, either to act on the beam itself, or to adjust measurement devices to obtain beam properties. This is done through "knob-like" tools which can be attached to any parameter, providing incremental actions, on/off type action, and several other moves, for ex. return to initial value. The refreshed acquisition value is also displayed.

Individual controls are also concentrated in a *parameter list* table which gives a complete information on status, control and acquisition values relevant for this Application area. Any line of this list is a selectable object calling a control knob.

This list is similar to the Synoptics, but gives other pieces of information: in that way List and Synoptics are complementary instruments.

The two classes above are entirely valid in any Application: they are what we called "generic application programs" [6].

Remark: we do not deal here with the question of data management (database use) which is entirely a part of the Control Systems specifications.

Physical Parameters

Some hardware variables like Voltage, Intensity... are hidden. So an accelerator physicist can adjust parameters directly in normalized position units, meters, radians, gauss, corresponding to the physics involved in the process [7]. Generally the two types of variables are linked by a linear equation system seen as a matrix which can be inverted to play in both directions. Going further this way [8] we can treat the synchronization of pulsed process (here the CPS 1 GeV Injection) with this method, the matrix being the cabling lay-out.

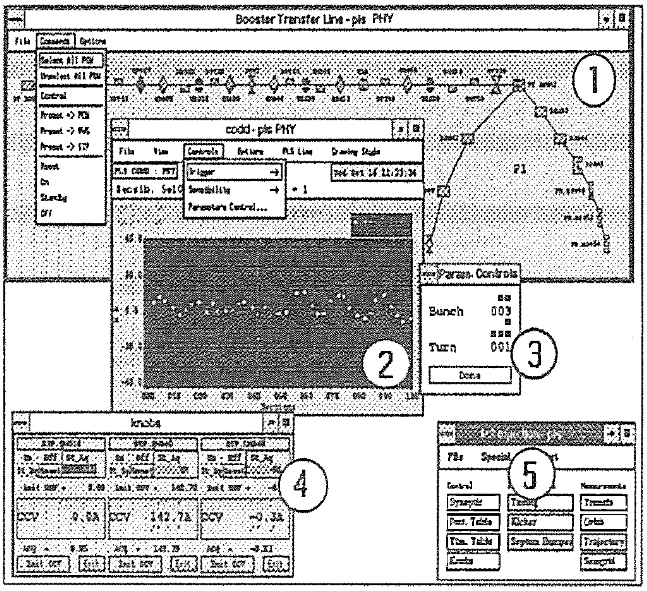


figure 2: one example of a Workstation screen showing some of the tools and presentation used for the PSB-CPS 1 GeV transfer and injection prototype.

1= the Synoptics window including all sub process elements, with Controls menu
2= Closed Orbit Display, in graphical view, and its Controls pull-down menu
3= the Closed Orbit Display sub window with the synchronization commands
4= three (5 possible) "knob-like" tools, controlling power supplies (hardw. units)
5= the main window - Application manager - giving total access to all programs

Measurements. Complex Treatments.

This class of tools has no special structure, being too linked to the particularities of the process. Nevertheless, two types of measurement tools are distinguished :

- *special measurements:* the apparatus used, the diagnoses, the parameters involved are present only in the process covered by the Application. These tools are developed (sometimes using complicated computations) according to specific needs. They only obey the presentation rules: e.g. the Transverse Emittance measurement.

- *dedication of general interest measurement:* here the measurement is performed through a tool with conditions related to the Application (synchronism, sensitivity, ..etc..) from which the tool is called. Then, this measure is "generic", i.e. usable from several Applications but each of them sees it as if it was specially written. In the prototype, the Closed Orbit Display serves as an example [9].

One other class can exist:

Optimization

Of course, these programs are specially designed for an Application; each of them depends only on the process analyzed. In the prototype the Betatron Oscillations Minimization [8] was developed; others are foreseen. There again no standards are anticipated, only the interactions and presentation rules are adopted; the only decision taken was to propose computed corrections to the user : no automatic closed loops were introduced.

These five types of tools will be present in any Application of the man-machine interface. Their relative weight will depend on the sub process characteristics.

IV SCREEN MANAGEMENT

The different tools from the 5 classes above constitute a working environment. The user is facing a Workstation screen: we had to define some standards of data presentation which could be used in the whole control interface [10]. We are not talking here about the Standards selected for the Computer Controls System: for screen picturing, X-Windows™ and Motif™ were used (see [6]) and their standards accepted.

Data presentation

We concluded from past experience that in any display exists a dichotomy of data presentation: graphics or tables . Due to the user personality, or the type of data, or the mode of operation ...etc.. it is difficult to eliminate one of them. We tried to keep at will the two displays (example: Closed Orbit Display). The *synoptics* and *parameter lists* are themselves applications of this concept.

Windows use

Each type of controls tools is running in a proper window with a menu bar and its associated pull-down menus. We intend to propose at this level a common choice which can be enlarged according to the sub process. In the prototype FILE, VIEW, CONTROLS, OPTIONS.. HELP are the common factor; it is too early to freeze the proposal, but the subject is important.

Another point of interest consists of the windows movements. Experience has shown that windows must be opened at the same place every time they are called, and, hardly ever need re-dimensioning. It is a task for a man-machine interface study to define an Application in order to verify that this request is satisfied. If necessary, the freedom of re-dimensioning a window is not given. Whenever possible sub windows should not mask present windows.

Icon use

Such facilities are powerful tools for screen management. Here also, to make an efficient use of time in an Application we imposed some rules, and introduced a hierarchy on the icon transformations.

With the exception of some windows waiting for a specific response or being the last of a tree structure, each window can be put into an icon. The latter is kept itself in a window (bottom of the login screen) and can be re-opened to its full scale. By the way, we decided also that no window should occupy the whole screen.

When opening an Application, a main window is opened and proposes the list of available programs in this sub process. At any time, even with a screen full of running displays, you can clean the entire screen by storing this main window into an icon . And the reverse action puts back on the screen all your Application working environment, exactly as it was, in one go. This facility has been found extremely useful, allowing a fast glance at another subject without interrupting our main task.

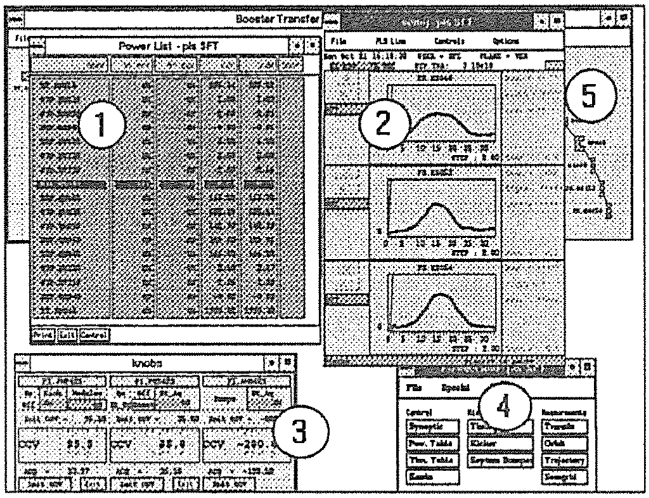


figure 3: another Workstation screen including tools showing other views and means of controls associated to the PSB-CPS 1 GeV injection process, and complementing the figure 2. All these tools can be activated together or successively by the way of icons.

1= Parameter List of the power supplies, giving complete numerical information
2= the Beam Profile Measurement window and related Emittance computation
3= several knobs, linked to normalized coordinate units here (physical units)
4= the main window - Application manager - see figure 2
5= the Synoptics view kept in background, e.g. giving instantaneous status info.

Workstation use

Every Workstation will be independent, see Chap. V.

It is foreseen that only one Application will be opened (i.e. "alive") at the same time on a Workstation screen. This is not introduced as a rule, but a suggestion. The hierarchy principle implemented in icons use will ease the fast exchange between several Applications on one Workstation.

Updating data

As said in Chap.I, an Application is related to an accelerator cycle providing a user with a beam; it was requested that all the programmes belonging to an Application display a set of coherent data, i.e. presenting values acquired on the same machine cycle. At least a warning sign must indicate clearly if it is not the case for whatever reason. In principle, several Applications could co-exist on a Workstation, and updating rate problems could be anticipated. This is why we considered that probably only one Application will be "full screen" at a time on one Workstation.

V ANTICIPATED CONTROL CENTER

Workstation relationship

Every Workstation will be independent, i.e. can open any Application (as defined in Chap.I), and there will be no mutual interaction between any other Workstation. If needed, several Workstations will (the system will allow it) open the same Application in order to get a larger screen surface. Controls will be granted on the basis of first in - first served, and the command values will be refreshed with the last updating (see [9] more precisely).

Numbers

The prototype gave us an idea as to the needs arising from the operation of an accelerator complex, and how the present central control room - from which ten machines are driven - will be rejuvenated. Thanks to the non-dedication of any tools (the present consoles), the PS accelerators complex has been operated from 8 simultaneously active entry points for several years.

For reasons of screen surface we concluded from experience that we need 3 stations to replace one console, making what we call a "work place". For the time being, the general analog signal multiplexing system will be kept; however it is foreseen [11] in the Computer Control Rejuvenation project to construct a digital system using Workstations as display terminal.

We think that the central control room of the PS complex will include at least 25 Workstations, with a mean value of three logged on one accelerator at a time. One extra improvement will be to increase by a factor of 3 the number of entry points.

Assignment

At a work place, it is foreseen that two workstations will support the Applications defined here, and the third one will be mainly used to cope with the generalities: Alarms, Messages, Radiations Survey, Statistics, and so on.

VI CONCLUSION

The Man-Machine Interface specifications summarized here and introduced in this prototype are not final. Other important points are still being debated. For instance spreadsheet techniques (like the EXCEL™ product available in our Personal Computer network [12]) could replace some development programs if not operational tools. WINGZ™ is currently under evaluation [9].

On the other side a few topics were left aside but will have to be defined correctly before the user environment definition will be abandoned. Designing the Data Archiving and Error Handling and Presentation systems is a must.

For several months the Booster-to-CPS transfer and injection process at 1 GeV has been operated through one Application, built from the specifications given above. This Application constitutes a global environment providing the users with the controls, analysis, visualization of a part of an accelerator. This was defined as a prototype for the new era in man-machine interface using graphical displays associated with a distributed network of powerful workstations. The fact that this prototype was readily accepted by the operation teams without a long training period encourages us to extend the principles used. A second beam line is already treated in the same way.

The PS complex Computer Controls Rejuvenation project will now take care of an ensemble of three machines: the LEP pre-injectors (2 Lepton Linear accelerators + 600 MeV collecting ring) [13]. A large part of the prototype programs will be used for that slice, to the benefit of users.

VII REFERENCES

[1] F.Perriollat and C.Serre, "Problems in Rejuvenating a Control System and Plans for the CERN Proton Synchrotron (CPS)" Nuclear Instruments and Methods in Physics Research A293 (1990) pp. 81-86.

[2] PS and SL Control Groups editors, "PS/SL Controls Consolidation Project", Technical Report, CERN Internal rep. PS/91-09(CO) - CERN SL/91-12(CO) April 1991.

[3] R.Rauch and Ch.Serre editors, "Common Control System for the CERN PS and SPS Accelerators", these Proceedings (ICALEPS, Tsukuba, Japan Nov. 1991).

[4] J.Boillot, M.Boutheon and al. , "Operation oriented Computer Controls for the CERN-PS Complex", IEEE Trans. on Nucl. Science, Vol. NS-28, nr 3, p.2261 (1981).

[5] J.Boillot, G.Daems and al. , "Pulse-to-Pulse Modulation of the Beam Characteristics and Utilization in the CERN-PS Accelerator complex", IEEE Trans. on Nucl. Science, Vol . NS-28, nr 3, p.2191 (1981).

[6] P.Antonsanti and al. , "Workstations as Consoles for the CERN-PS Complex: Setting-Up an Environment", these Proceedings (ICALEPS, Tsukuba, Japan Nov. 1991).

[7] J.Boillot, G.Daems and al. , ibid.

[8] M.Arruat and al. , "New Controls for the CERN/PS Hadron Injection Process using Operating Tools and High-Level Modelling Accelerator Programs", these Proceedings (ICALEPS, Tsukuba, Japan Nov. 1991).

[9] P.Antonsanti and al. , ibid.

[10] M.Boutheon and A.Pace, "A Proposal For a General Man-Workstation Interface used in Accelerators Controls", Internal report PS/OP/Note 90-10, March 1990 (contains a large Reference list).

[11] B.Frammery, " Proposal for a VXI digitized Signal Observation System for the PS Complex", Intern. rep. PS/OP/Note 91-30, Oct. 1991.

[12] A.Pace, "Office Local Area network at the PS Division", Intern. rep. CERN/PS 91-45 (OP), Sept. 1991.

[13] G.Benincasa, and al. , "Rejuvenation of the CPS Control Systems: the First Slice", presented at the Particle Accelerator Conference 1991 May, San Francisco, Internal rep. CERN/PS/91-14(CO) April 1991.

The Replacement of Touch-Terminal Consoles of the CERN Antiproton Accumulator Complex (AAC) by Office PC's As Well As X-Windows Based Workstations

V. Chohan, I. Deloose, and G. Shering
CERN, PS Division, CH-1211 Geneva 23

Abstract

With aging hardware and expensive maintenance and replacement possibilities, it was decided to upgrade the AAC touch terminal consoles with modern hardware. With significant amount of operational application software developed with touch terminals over 10 years, the philosophy adopted was to attempt a total emulation of these console functions of touch actions, graphics display as well as simple keyboard terminal entry onto the front-end computer controlling the AAC. The PC based emulation by mouse and multiple windows under MS-DOS and later, under the Windows 3 environment was realized relatively quickly; the next stage was therefore to do the same on the Unix platform using software based on X-windows. The communications channel was established using the TCP/IP socket library. This paper reviews this work up to the operational implementation for routine control room usage for both these solutions.

INTRODUCTION

The CERN Antiproton Accumulator Complex (AAC) is composed of two circular, concentric ring accelerators and an antiproton production area (see Fig. 1). The inner ring, the Antiproton Accumulator (AA) was commissioned in 1980 while the outer, Collector ring (AC) was brought into operation in 1987 to permit an order of magnitude increase in the antiproton flux. The AA was conceived initially as an experiment and was built and commissioned in record time while the CERN PS Complex of accelerators was undergoing major changes from rudimentary to modern computer controls. For reasons of time and financial expediency, it was considered necessary to have cheap operator interaction means available for the AA commissioning, with simple to use interpreter (Nodal) based facilities. The Touch Terminals [1, 2], developed and used for the CERN-SPS control room were ideally suited for this role [3]. The controls system provided the necessary facilities to connect the Touch Terminals to the equipment. The AA controls system and its extension and upgrade in 1986 has been amply described elsewhere [4, 5].

THE PRESENT TOUCH TERMINALS

The Touch Terminal (TT) is a specially configured mini-CAMAC crate with a microprocessor and special modules to drive a touch button screen, a graphics and character display screen and is connected to the front-end computer which controls the equipment via CAMAC Serial highways. Communication between the computer and the TT is by means of the standard current loop serial interface. The microprocessor controller in the TT is programmed to be transparent to the front-end computer terminal driver. However, it also detects or inserts certain "escape sequences" enabling the simple touch button functions like LEGEND, BUTTON etc. and graphic monitor functions like VECT, TEXT and so forth. Hence, the TT simply appears as a standard terminal to the controls computer but provides powerful interaction facilities with equipment. For the antiproton improvement programme at CERN and in preparation for the construction/commissioning of the AC ring in 1986-87, the TT's were upgraded to a Motorola 68000 based microprocessor, permitting colour alphanumeric and graphic facilities as well as higher terminal speeds. This, together with a faster front-end computer, has permitted up to five operational TT's for the AAC since 1986.

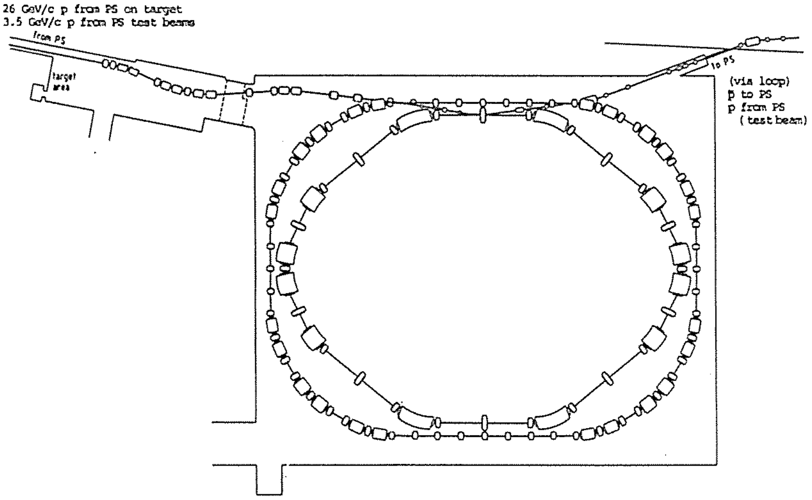


Fig. 1. General layout (magnetic elements only) of the Antiproton Accumulator Complex (AAC): outer ring - Antiproton Collector (AC), inner ring - Antiproton Accumulator (AA).

GOALS AND NEEDS FOR TOUCH TERMINAL REPLACEMENT

In the years 1988-89, it was increasingly clear that the mini-CAMAC crate based TT's, although cheap compared to the conventional fully-fledged minicomputer based operator consoles, were at least a factor three more expensive than a modern, commercial office Personal Computer (PC). With limited spare hardware and expensive repair/maintenance possibilities, it was obvious that the TT's, thanks to the simplicity of usage and direct connection to the front-end computer, could be replaced by a powerful PC using a standard VGA or super-VGA (1024 × 768 pixels) graphics card and ethernet, TCP/IP links.

With vast and running investment in thousands of lines of application software and very limited annual accelerator shutdown time (< two months every winter), the primary goal was to literally emulate the complete TT facilities on a PC, using multiple windows to provide the pseudo-touch (click by mouse on touch area), graphics window and the terminal keyboard echo window. The accelerator dependent applications suite of programs [6] and automated processes in the AAC represent a large number of man-years of software refinement and effort and CERN's ongoing physics programmes did not have the resources to make any extensive alterations or modifications; hence the original applications code had to run on the new PC-based TT as well as being transparent to the old TT's at the same time. This cohabitation of the new with the old and a graceful transition during the normal accelerator running (>6000 hours/year) was an important issue that precipitated the idea of a fully-fledged TT Emulator.

At the same time, the CERN management had mooted the idea to dismantle the AAC and re-assemble it in the USSR at the UNK complex, with a view to collaborate and continue the proton-antiproton Collider programme in the TeV range. It was considered essential that the hardware and software be maintainable for several years if this move did occur [7]; the PC-based Emulator fitted this criterion ideally.

With the adapting of the PS Complex controls system to industry standards and trends at least at the operator interaction level, the TT Emulator also needed to work under a modern RISC architecture workstation, running UNIX, X-Windows and MOTIF tool kit. In this manner the AAC could converge to the same chosen standard interaction means as the rest of the PS complex as well as SPS-LEP. Hence an ultimate aim was also to be able to do this, based on DEC-3100 workstations.

EMULATOR DESCRIPTION AND PC-BASED FACILITIES

The Emulator work commenced in early 1990 using an office Olivetti 386 PC running MS-DOS as a target system. The PS Division has a network of office PC's, connected via ethernet and the controls computer are also interconnected via ethernet and TCP/IP protocols to the office network. Figure 2 illustrates a simple schematic layout of this connection to the AA front-end computer controlling both the AA and AC rings. The Emulator task was developed such that it automatically establishes a two-way socket stream (telnet) to the AA computer at start-up, using the low-level suite of TCP/IP socket library routines. Immediately afterwards, remote log-in is carried out and the necessary procedures are automatically established to access real-time Nodal facilities, permitting full accelerator control. This then establishes the identical environment under which the existing TT's run. Subsequently, the Emulator creates the three windows on the single PC screen emulating the three screens of the TT namely, the touch screen, the graphics/alphanumeric display screen and the terminal VDU. Having established the communications, log-in and respective emulated windows, the main function of the emulator reduces to the correct detection, interpretation, redirection to the right window and insertion of byte streams and 'escape sequences', just as was done by the TT microprocessor. An essential difference of course was that the created pseudo-touch buttons on the touch window have to be clicked by the mouse; this and other features like the moving, re-sizing and automatic re-scaling of every window and provision of essential services like the hard-copy printout of any window, graceful exit, etc meant that a continual scan of the keyboard, mouse and communications input buffer (TCP/IP) is necessary in the main loop of the Emulator. Initially, the DOS-based Emulator was developed in Microsoft 'C' and used a self-written graphical window manager for this application. This has been replaced to run under MS-Windows 3.0, using the standard tools available under Windows 3.0 and including the Software Development Kit (SDK) for Windows. The Windows version permits increased flexibility and compatibility with normal Windows 3.0 applications and uses interrupts for keyboard and mouse instead of a continual scan. Using a high resolution graphics board and a larger screen, two concurrent Touch Terminals can be run on the same PC. Figure 3 shows the schematic Emulator layout in the current PC environment and the respective links. Figure 4 illustrates a typical PC three-window Emulator screen dump, as used by an accelerator application program.

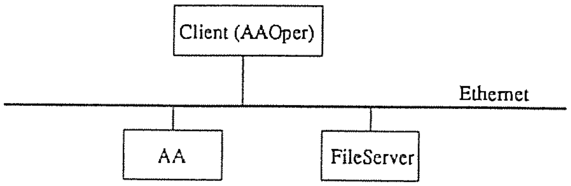


Fig. 2 : Network layout

FACILITIES ON DEC-3100 WORKSTATIONS

The success of the PC-based Emulator augured well for next stage of the project to have the same facilities on a DEC-3100 workstation running Ultrix. The PS Complex has

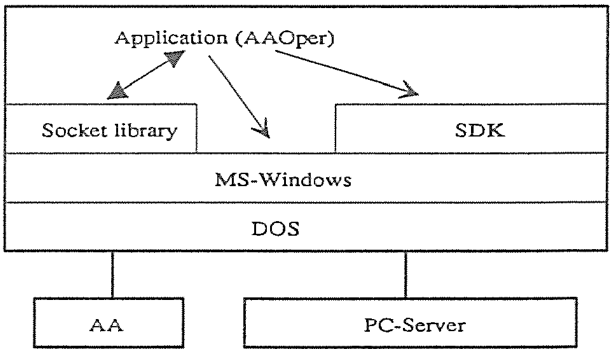


Fig. 3 : Software levels for Windows 3 (PC)

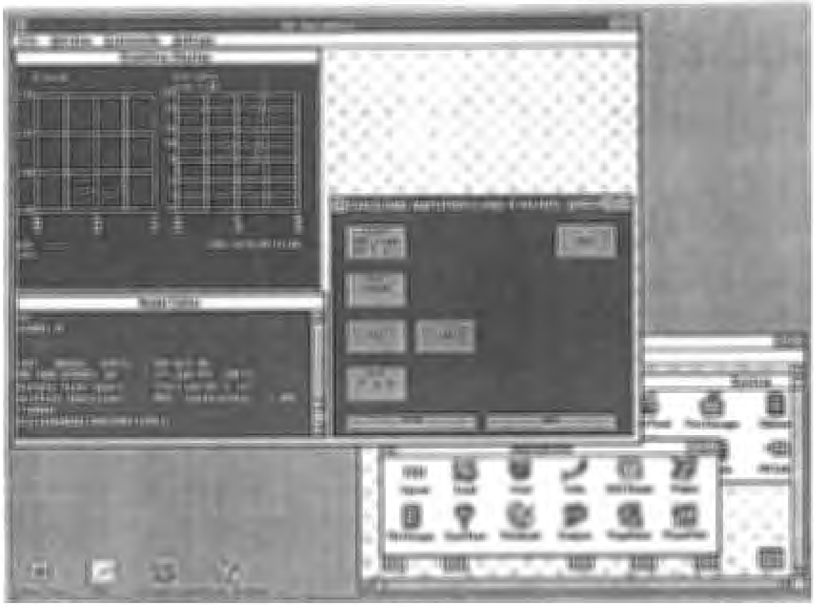


Fig. 4. Emulator running under Windows 3 on a PC.

standardized on this hardware recently, together with the X-11 library and MOTIF tool kit at the intermediate level. The Emulator was specified to use the full client-server relationships based on these protocols on top of the TCP/IP socket library implementation for the communication link as in the PC case. Figure 5 shows a schematic layout and interconnection of the Emulator software on a DEC-3100 workstation. The DEC-3100 workstation provides increased speed due to the RISC architecture; it also has a larger screen as a standard compared to a normal office PC, providing high resolution, convenience and ease-of-use for the three window TT emulation. Much of the low-level source code was ported from the PC-Emulator to the DEC-3100. Figure 6 shows the screen output of the complete three-window emulator running an operational application program on DEC-3100 workstation.

CHARACTER AND GRAPHICS FACILITIES

The Emulator package uses the original definitions for the character and graphics facilities as were defined for the TT [8, 9]. In the TT, the high-level commands to draw characters, vectors, circles, polygons, etc. are broken down into individual dot commands which are then output to the display through the Display Memory (DIME) module. Hence, for the Emulator, the essential task was to convert the character and graphics escape sequences into C-language based calls for both the PC or the DEC-3100. The major difference is due to the fact that since multiple windows are needed for creating the pseudo-touch panel, graphics display and the terminal-echo, respective memory buffers are required and used at every instance such that correct re-size, re-direction or hide/expose events can take place for each window. For the graphics window, separate memory buffers are required for both the graphics and text information. The graphics display continues to support a resolution of 768 × 576 pixels while for the character display, the standard size text is supported for 24 lines with 64 characters, reduced to 12 × 32 for large size text. Full technical details of all the definitions and additional features of the Emulator are given elsewhere [10].

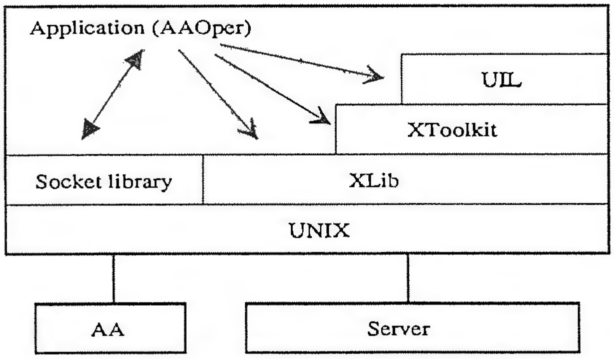


Fig. 5 : Software levels for X-Windows (Unix)

PORTING APPLICATION PROGRAMS AND PERFORMANCE

The whole suite of accelerator dependent application programs and procedures for the AAC Complex have undergone the ultimate trial in usage from both the PC or the DEC-3100 workstations, without needing any changes. In fact, the original three TT's in the local control room have not been decommisssioned from use, hence the *de facto* cohabitation is an absolute necessity. Maintenance of a unique set of accelerator application programs limits the software interventions to this level only, independent of the three variants (TT, PC or DEC-3100) possible at the operator interaction level. The application programs have been routinely used for accelerator operations by the shift crew on both the Emulators. While there may be little difference between a PC or DEC-3100 based Emulator, the operators have experienced a considerable improvement in speed over the old TT's. The aspects of window hard-copy and multiple window printouts on a single page have provided an improved facility, highly appreciated as a paper-saving, ecological solution.

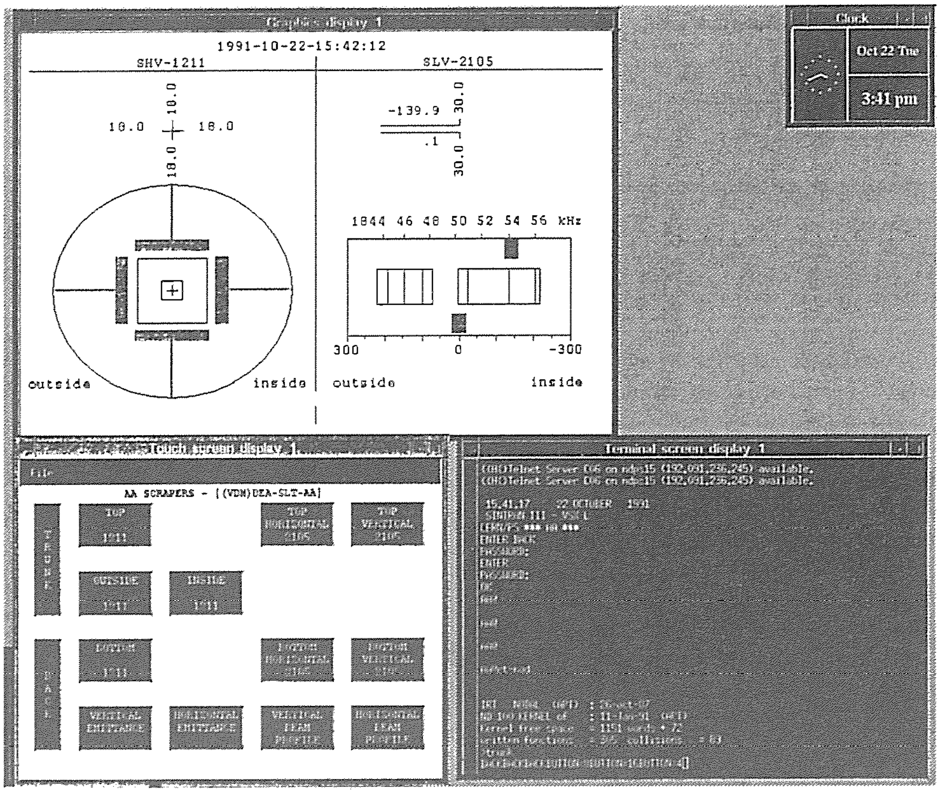


Fig. 6. Emulator running on a DEC-3100 Workstation.

While the Emulator benefits from the modern ethernet TCP/IP links, the response time is dependent to a large extent on the front-end Norsk-Data controls computer; however, for the final display features and facilities, the modern graphics hardware and software provide a large factor of improvement in speed. Overall, the net effect is a factor three gain in speed over the terminal connection TT's in the main control room and factor ten gain in the local control room.

CONCLUSIONS

The Emulator has been put into routine operation using both the DEC-3100 and the Windows 3 PC environment. There has been a total and welcome acceptance of these facilities by the operations crew without any great need for additional training on usage; the latter aspect permitted installation and usage even during normal accelerator runs for physics.

Within the usual accelerator shutdown constraints and planned hardware maintenance, upgrades and spending profiles over several years, the Emulator has provided a tremendous bonus in permitting multivendor modern hardware and software to be successfully introduced in parallel with the partial and graceful de-commissioning of existing, aging TT's. The ethernet links across the accelerator laboratory and offices permits further advantage in ease of accelerator supervision and initial trouble-shooting, without recourse to urgent visits to the control room.

For the future, since the Emulator package permits the use of up to two concurrent consoles on the same, larger screen workstation, a significant amount of flexibility and substantial saving in hardware is possible in the local control room; a graceful de-commissioning of the three old-style TT's has been planned over the next two years, to be replaced by a pair of Emulator workstations, each running two concurrent consoles.

REFERENCES

[1] P. Anderssen et al., "The Touch Terminal", CERN/SPS/81-15 (AOP), Proc. of the Topical Conf. on Application of Micro-processors in High Energy Physics Experiments, Geneva, Switzerland, May 1981.
[2] G. Shering, *Proc. of Europhysics Conf. on Computing in Accelerator Design and Operation*, Lecture Notes in Phys. 215, Springer 1984, p.481.
[3] G. Shering, "The Touch Terminal for AA", CERN/PS/AA Mem. 80-10, 1980.
[4] D. Blechschmidt et al., IEEE Trans. Nucl. Sci., NS-28, 1981, p. 2258.
[5] V. Chohan, *Proc. of the Int. Conf. on Accelerator and Large Experimental Physics Control Systems*, Vancouver, BC, Canada, 1989 , Nucl. Instrum. Methods, A293, 1990, p. 93.
[6] V. Chohan , S. Van der Meer, *Proc. of the Int. Conf. on Accelerator and Large Experimental Physics Control Systems*, Vancouver, BC, Canada, 1989, Nucl. Instrum. and Methods, A293, 1990, p. 98.
[7] V. Chohan , "The Future of AAC Controls System: Options for Discussion", CERN/PS/AR/Note 89-8, August 1989.
[8] P. Anderssen, "Programming the SPS Consoles", CERN SPS/82-1 (ACC)
[9] P. Anderssen et al., "User Manual for the New 68000 DICO" CERN/SPS/ACC-CC/Note 82-42, July 1983.
[10] I. Deloose et al , "User Manual for the PC and DEC3100 Workstation Based Touch Terminal Emulator", CERN PS/OP AR Note (to be published).

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI04

The Elettra Man-Machine Interface

Franco Potepan, Giuseppe Surace, Marco Mignacco
Sincrotrone Trieste
Padriciano 99, 34012 Trieste, Italy

Abstract

ELETTRA is a third generation Synchrotron Light Source under construction in Trieste (Italy), with beam energies between 1.5 and 2 GeV. Two networks connect three layers of computers in a fully distributed architecture. An ergonomic and unified approach in the realization of the human interface for the ELETTRA storage ring has led to the adoption of artificial reality criteria for the definition of the system synoptic representation and user interaction. Users can navigate inside a graphic database of the whole system and interactively edit specific virtual control panels to operate on the controlled equipment. UNIX workstations with extended graphic capabilities as operator consoles are used in the implementation of the PSI (Programmable Synoptic Interface), that was developed on top of X11 and PHIGS standards.

I. INTRODUCTION

We may think of an ideal man-machine interface (MMI) as a tool which allows users to interactively specify their preferred means of interaction with the devices controlled, using graphic programming techniques to compose predefined objects that support the exchange of numeric, string or graphic information. A definition of new specific objects is also possible using an editor with similar properties.

Commercial interfaces ado not generally combine these two requirements without the introduction of a considerable amount of new concepts and notions. Today we can profit by the wide diffusion of some advanced graphic user interfaces (GUIs), based on a well defined set of composable objects, or *widgets*, that have certainly increased the knowledge of interface principles and interaction paradigms among users. A similar awareness at the lower level, where graphic primitives and basic interaction techniques should be structured together to form an object, is at present unthinkable. As a matter of fact, a global definition of the semantic and syntax of graphic human-computer interaction still suffers from a lack of standardization of the relative graphic and dialogue lexicon, which is a young field of active research. Interactive widgets composers are indeed filling up the market, while a similar approach for the definition of widgets is practically abandoned.

II. DESIGN GOALS

In our project of a MMI for the Elettra storage ring the aim of obtaining a well balanced integration between new powerful features available on modern hardware and software platforms, and a comprehensive exploitment of innovative methodologies concerning human-machine interfaces, has led to a continuous revision of the design and implementation phases, together with a considerable effort in testing prototypes with real users.

The usual top-down design approach which uniquely guarantees consistency in terms of colour coding, menus and dialogues layout, warning messages and overall behaviour of the user interface is left at the end, when all possible user interactions are already defined.

Two main lines have been followed in the design of Ψ (Programmable Synoptic Interface), in order to keep the amount of new concepts required for its operation to a minimum:

- to hide all the details concerning the specific structure of our control system from the users. A complete transparency of operative system, programming language, graphic and communication libraries is therefore essential.

- to take full advantage of all the notions users are already familiar with: the local operations of the devices, the planimetrical layout of the machine, the commonly used desktop metaphor as a computer interface.

III. CONCEPTUAL ORGANIZATION

A possible solution to these demands was to adopt typical artificial reality criteria for the design of our interface. A planimetric representation of the whole system, where all the items controlled are shown with their real shape and position, was recognized as the users' most familiar environment. The equipment displayed inside the environment is associated to virtual control panels which allow users to operate the graphic representation of a large set of devices like switches, knobs, sliders, digital and analogue indicators, whose behaviour is equivalent to that of the same instrumental devices. A set of well defined interaction paradigms regulates the navigation inside the environment, the modification of the scale and visibility of the layers into which the graphic information is structured, the selection of devices and the activation of the relative control panels.

Let us now distinguish between the principal elements that compose our artificial world:

-an *environment*, i.e. a synoptic representation of the whole system;

- a set of *objects*, i.e. the devices controlled;

- a collection of *virtual control panels*, i.e. the logical grouping of controls associated to one or more devices;

- a set of *interaction paradigms* between the user, the environment and the objects.

The environment consists of a complete graphic database of the whole system, no longer restricted to fixed size views of selected parts of the plant. If we define the graphic database at system level, freeing the user from the synoptic drafting phase, we can profit by the centralized organization of the data, exploit all the advanced features of the adopted graphic system and make graphic conventions and interactions rules consistent and comprehensible.

Objects are the graphic representation of the controlled devices. They retain, as far as possible, their original planimetrical shape and location, without any additional graphic coding. They are quite similar to the other graphic primitives inside the environment, like buildings and planimetrical references, but they can also be selected and activated, and their representation can be changed according to their state.

Each object is associated to one or more control panels, within which users can operate on the virtual controls to modify the values of a specific controlled device.

Interaction paradigms vary between the different parts. Users can interactively navigate inside the environment, specifying the position and the size of the view. A hierarchical organization into layers of the visualized data and a well defined set of properties can easily overcome the high density of information inside the environment. Layers can be selectively visualized, while further detail and information in text form can be automatically obtained as a threshold function of the scale; logically related objects can be highlighted or changed in scale to improve readability.

IV. IMPLEMENTATION

The Elettra Control System has a fully distributed architecture, organized into three layers of computers: control room, local process computers and equipment interface units. Two networks interconnect the adjacent computer layers. At control room level UNIX workstations are used as operators' consoles [1]. This architecture, however highly adaptive to future developments, implies the adoption of heterogeneous hardware and software between the two networks. Whereas the transparency of the lower operative system is guaranteed by the Remote Procedure Call application protocol [2], naming conventions and servers location among the process level computers are still required at console level.

We have adopted the X Window System Version 11 as our windowing system and basic graphic library, together with the OSF/Motif window manager, Xt and Xm libraries [3][4]. The Motif widget set had to be extended with several home-made widgets to fulfil typical accelerator control requirements. The wide application of this GUI to accelerator controls has led to an active collaboration and exchange of information between several institutes. This might be a good starting point for a definition of common requirements.

Although X11 includes a reasonable number of routines for creating basic graphic primitives and for modifying attributes, it is definitely limited as far as display-list organization and mass storage is concerned. It is mainly a bit-mapped oriented library and the download of visualized portions of graphs is the only storage method available. The lack of any form of data structure within a display list in memory and the consequent impossibility of recognizing a single graphic primitive within the display, of rotating, translating or resizing it, confirmed the inadequacy of this library for the synoptic representation of our storage ring.

We therefore opted for the adoption of PHIGS (Programmers' Hierarchical Interactive Graphics Library) for vectorial graphics. Actual implementations of this standard (ISO-IS in 1988) move towards its complete integration into the X11 graphic interface and event handling merging [5][6]. The PEX (Phigs Extended X) extension of the X protocol, now appearing in its first implementations, provides the transfer of both bit-mapped and vectorial primitives over the network. The advanced display list organization of Phigs into a hierarchical tree structure of primitives [7] perfectly matches our requirements for the centralized database. Moreover, this standard supports unique definition of primitive attributes that includes transformation matrices, in addition to all common vectorial libraries features like archiving and single primitive picking. Maximum performance in different hardware platforms is possible as all primitives suitable of acceleration are downloaded to the graphic subsystem, as specified in the standard. Nowadays, typical graphic acceleration of 1M vectors/s can be easily gained with no optional graphic subsystem. The constant trend of hardware makers towards the adoption of powerful 2D or even 3D graphic accelerators as the default configuration of their workstations, guarantees the acquisition of advanced graphic features, such as real time navigation over a graphic database, with no additional cost.

A. The graphic database

The naming convention adopted in our system is both simple and effective. Any device is defined by its four component string, which specifies the *family* to which the device belongs, the *member* within the family, the type of *action* requested (current read, fault reset, etc.), and the *mode* in which data is transferred (read real, write integer array, etc.) [8].

The main effort in the definition of an exact representation of the storage ring was actually the drafting of all the components in real size through a graphic editor. As this task is always accomplished using common CAD programs, and the machine assembly needs an identical database, we thought of importing the data from our CAD stations via a filtered download of the graphic primitives. The only modification to the CAD database was the addition of a label to each object, specifying its family and member. The link to the control system database was therefore obtained. We imported the pre-existent organization into layers, providing a logical separation between the various parts of the machine, and added the hierarchical structure respecting our naming convention, inserting a further "group" node between layers and families.

We developed a parser that translates Autocad DXF format files and generates a Phigs CSS (Centralized Structure Store). This parser recognizes the notion of blocks and layers and generates the tree structure shown in fig. 1. Buildings and other non-blocked graphic information are also imported and distinguished from the active objects by the absence of a naming label. The resulting structure is a tree of deepness five, ordered in root, layers, groups, families, members and objects.

We tested the good integration of Phigs into X11 defining a widget which loads a generic graphic structure from a file, and provides navigation and primitive picking with no other parameters than the filename inside the Xt widget creation call.

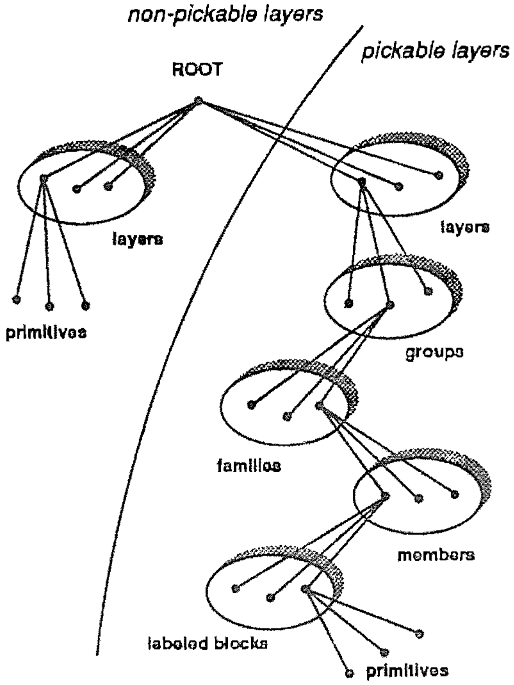


Figure 1. Hierarchical organization of graphic primitives inside the environment.

B. The Control Panel Editor

The time spent in the generation of the code required for a typical composition of widgets inside a control panel was estimated to be excessive when related to the number of panels to be provided to the users. A fundamental problem was also the definition of the types and positions of the widgets chosen for the control of a specified device. The possibility of adopting further application packages on top of those above listed has been carefully investigated. Several programs have therefore been examined in our and in other institutes [9]. Our opinion is that the integration with the underlaying standards is not yet acceptable, with the exception of widget-based interface generators, in which external data connection and home-made widget inclusion is still quite hard. As a result we decided to invest our time in the development of a fully interactive Control Panel Editor (CPE).

We shall distinguish two operative modes from now forward: an edit mode, in which we use CPE to create or modify the panels associated to an object; a run mode, in which we select an object, activate the respective panel and act on the correspondent controls (fig. 2).

The basic interaction tasks (BITs) [10] used in selecting, positioning and resizing widgets inside a control panel are strictly those specified in the OSF/Motif Style Guide document, and are almost identical to those used in other well known GUIs with the same desktop metaphor. The possibility of interactively change selected properties of the widgets that are being edited allows users to have an immediate feedback as the property is changed. We deliberately limited the number of modifiable properties to a definable set, avoiding the change of those properties with an explicit coding, like colour, and of those with an unwanted effect to the interface, or an unknown meaning to the user.

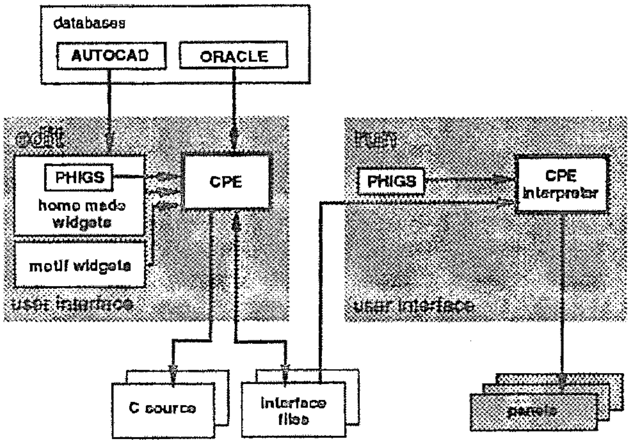


Figure 2. Schematic layout of the Control Panel Editor.

A complete transparency even to naming convention and server location of Ψ was possible with the integration of CPE with the graphic database. Introducing the notion of *association*, which is the only complex interaction task in our interface, we are able to specify a complete data connection with a device.

Let us follow the steps required to create an association. While in edit mode, we select the object we want to control from the synoptic. As the object is picked, its family and member information is used to access a list of all possible action and modes from the machine database. The selection of one of these items highlights all the widgets that are able to handle the specified type of data. As we place the chosen widget inside a panel, the association between the selected device and the widget is defined, and a connection to the relative equipment is established as soon as we enter the run mode and pop-up the panel.

Internally, CPE generates an interface file, listing all interactively selected properties in a purely declarative language. Interface files can be edited with a traditional text editor, enabling system programmers to quickly configurate Ψ with prototyped panels created cutting and pasting properties among interface files.

The generation of pure Motif C source code after panel composition is already available: a self-contained main program is created with explicit callbacks for each edited widget. Application programmers thus have an interactive tool for the rapid prototyping of the graphic and data connection parts of their programs. The implementation inside CPE of a small C interpreter, limited to the recognition of blocks of code, could overcome the limitation of the one-way use of the editor as a C code generator, and guarantees a complete development environment for application programmers.

V. CONCLUSIONS

The application of artificial reality criteria to controls is by no means original. Our main effort has been the implementation of these concepts on top of a standard software platform, obtaining a high level of portability among different hardware configurations. The modularity of the various parts of our interface (synoptic, panels, editor) and their consequent easiness of maintenance and upgrade, assures a longer life of the system, compared to the adoption of proprietary solutions.

VI. ACKNOWLEDGEMENTS

We gratefully acknowledge the support of all the other members of our group: L. Barbina, D. Bulfone and P. Michelini. We are indebted to Fabio Barbo for his valuable work in the definition of the whole graphic database. In particular we wish to thank Claudio Scafuri for his significant contribution to the design and development of our interface.

VII. REFERENCES

[1] M. Mignacco,*"The ELETTRA Control System"*, NIM A293(1990), pp. 44-49.
[2] P. Andersen et al., *"User Guide to the Network Compiler Remote Procedure Call"*, LEP Controls Note 97, CERN 1989.
[3] A. Nye, T. O'Reilly. X11 Release 4, vol. 1-5, O'Reilly & Associates Inc, 1990.
[4] OSF/Motif Style Guide / Programmer's Guide, Prentice Hall, 1990.
[5] HP-PHIGS Version 2.11 Reference Manual, Hewlett-Packard USA, Jan. 1991.
[6] TGS Figaro+ Rel. 3.0 Reference Manual, Liant Software Corporation, San Diego, Jun. 1991.
[7] T. L. J. Howard et al., *"A Practical Introduction to PHIGS and PHIGSplus"*, Addison Wesley, 1991.
[8] M. C. Crowley-Milling, *"Naming Conventions for the ELETTRA Components"*, Sincrotrone Trieste, ST/M-TN-88/13, Sep. 1988.
[9] Di Maio et al., *Report on the study group for the selection of a software package for synoptics*, CERN Internal Report, Oct. 1990.
[10] J. Foley, A. van Dam, *Computer Graphics principles and practice*, Addison-Wesley, 1990.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI05

Exploiting the X-Window Environment to Expand the Number, Reach, and Usefulness of Fermilab[1] Accelerator Control Consoles

Kevin CAHILL, James SMEDINGHOFF
Fermilab , P.O. Box 500, Batavia, IL 60510, USA

Abstract

The Fermilab accelerator operator workstation of choice is now the Digital VAXstation running VMS and X-Window software. This new platform provides an easy to learn programming environment while support routines are expanding in number and power. The X-Window environment is exploited to provide remote consoles to users across long haul networks and to support multiple consoles on a single workstation. The integration of imaging systems, local datalogging, commercial and Physics community's software, and development facilities on the operator workstation adds functionality to the system. The locally engineered knob/pointer/keyboard interface solves the multiple keyboard and mouse problems of a multi-screen console. This paper will address these issues of Fermilab's accelerator operator workstations.

I. CONSOLE HARDWARE

The accelerator console is built around a VAXstation color workstation running VMS and X-Window software. The VAXstation 3200, VAXstation 3520, VAXstation 3100/30, 3100/38, 3100/76 and MicroVAX II are in use as workstation processors. The displays have either 1280 by 1024 or 1024 by 768 resolution with 8 bits per pixel.

Network communication to other accelerator processors is via accelerator-control network (ACNET) software using either a locally designed token-ring card or an Ethernet to token-ring bridge. Communication for system management, accelerator clock, and other purposes is via DECNET and TCP/IP over Ethernet.

A single-screen console provides full functionality, but some control room users require multiple displays. X-terminals using TCP/IP and Ethernet provide these additional screens. The NCD-17c is the preferred X-terminal for this control system.

II. EXPANDING THE NUMBERS

The number of consoles connected to the Fermilab accelerators is rapidly growing. Where 20 consoles served Fermilab for 10 years and budgets were prepared for 50 new consoles, the demands for VAXstation consoles are exceeding that estimate. 33 VAXstation consoles are active with 24 additional VAXstations scheduled to be purchased by Spring, 1992.

The increased numbers are due to a variety of factors. Many requests for accelerator consoles were denied over the years due to high cost and difficulty of installation. The new console with symbolic debugging support is a productive rapid cycle development machine. Users want convenient access to accelerator information, often in their office.

However, greater numbers of consoles present larger demands on central services and front end data acquisition nodes requiring consequent upgrades in those areas and the introduction of application program time-outs and other measures to balance high accessibility with overall throughput.

As new models of VAXstations are announced and released, we tend to purchase machines with better performance and value. It is clear that a console supporting a development cycle has an excess of cpu cycles and network bandwith. Utilizing that excess power is possible by supporting multiple consoles on a single VAXstation.

A. X-server consoles

The software architecture of a console is relatively simple. A large shared memory and shared library, several manager tasks and user applications make up a console. Splitting the shared memory region into a global area and multiple console specific area provides the ability to run additional alphanumeric, graphic, and utility managers, and sets of user applications. An X-server console is obtained at the cost of an X-terminal for the window displays. Twenty X-server consoles are in regular use primarily in the offices of programmers and accelerator operations specialists.

---

[1] Operated by Universities Research Association for the Department of Energy

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI05

B. Long haul consoles

The X-server consoles were demonstrated on a variety of machines around the lab and at vendors' offices in nearby communities. Sun workstations, Macintoshes, and a variety of X-terminals were compared for capability, cost and performance. Experimenters understanding the capabilities of X, convenience, and travel cost savings requested control system access to allow test and development from their home institution. We currently support occasional long haul access from Pennsylvania and Texas. Peformance of a long haul X-server console is reported as somewhat slower than a local network connection, but acceptable. The suitability and appropriateness of long haul networks for remote control system access may be debated.

C. The numbers

With a full complement of 50 VAXstations, the ability to support three consoles on a single VAXstation, and the trend towards providing office accessibility to all programmers, the total number of consoles supported by this system is expected to exceed 100 within a few years.

III. ADDING VALUE

The tools and services of the accelerator console platform are evolving and expanding at a rapid rate.

A. Console library

The console subroutine library (CLIB) consists of more than one thousand routines available to application programmers. More than half of the routines in CLIB may be classified as value added services not practical on the previous memory limited platform. Programmers have available new data acquisition, file management, and screen management tools. Screen management routines include movable/resizable/stackable windows, scrolling windows, complex menus, dialog boxes, and plotting tools. In addition, many new services for support of tape recorders, sequencer management, error handling, and display of digitized images exist.

B. Datalogging

A distributed, circular datalogger runs on many of the consoles providing datalogging at rates up to 15 Hz (but typically much slower) for a few hundred devices, and plot retrieval rates of nearly one thousand points per second.

The distributed datalogger offers custom features not practical in the sole central datalogger such as dynamically changing collection rates. Current growth of the distributed datalogger predicts up to twenty five such loggers will soon exist, offering datalogging capability for 5000 devices.

C. Supporting development

Users develop applications programs in VAX Fortran or C. The Management Environment for Controls Console Applications (MECCA) is a locally written source code capture system that captures and keeps a public copy of the source code for each application program on the system. MECCA forces users to follow strict conventions controlling the location of include files and user, group, and system libraries. Other facilities exist to build user applications with little restriction for testing without affecting the production versions. These test applications are deleted from the system each evening.

D. Other software

The exploitation of commercial or Physics community software is keenly sought. Mathematica and PAW for example have been installed and demonstrated but have no current operational function.

IV. MULTI-SCREEN CONSOLES

A major requirement of the Accelerator Operations group was that control room consoles have multiple screens and support the knob and the set-in-the-countertop trackball present on the old consoles. The X-terminals which are used for the additional screens come equipped with a keyboard and pointing device. Operating a console with multiple keyboards and pointing devices is unwieldy. This problem, coupled with the strong demand for a physical knob and non-standard pointing devices, resulted in the development of the locally engineered knob/pointer/keyboard box.

A. Control Room Console Configuration

A fully equipped main control room console consists of four physical screens, one 19 inch color VAXstation and three NCD 17 inch color X-terminals. The VAXstation screen contains four alphanumeric windows in support of the user's applications, and the utility window. One X-terminal displays the alarm window. The remaining two X-terminals contain the six graphics windows.

B. Knob/Pointer/Keyboard Box

The knob/pointer/keyboard box interfaces the knob shaft encoder, preferred by the operations staff, to the console. It allows a single keyboard and pointing device to provide key input and pointer motion to any of the four display devices. Finally, it supports the Digital and NCD mice and a variety of trackballs as pointing devices. No user consensus can be obtained on the best pointing device. Consequently, up to four pointing devices are ORed together into one logical pointing device which is sent to the selected screen. This allows for left and right-handed trackballs or a variety of pointing devices for the operator. As the pointer reaches a screen edge, software on the VAXstation instructs the knob/pointer/keyboard box to switch the pointer and keyboard to the adjacent display screen.

The box accepts inputs from the following devices:
- Digital keyboard (RS423, Digital protocol)
- Digital mouse (RS232, Digital protocol)
- Mouse-Track trackball (RS232, Digital protocol)
- Set-in-the-countertop trackball (pulse-train)
- NCD mouse (RS232, Logitech protocol)
- Knob shaft encoder (pulse-train)
- VAXstation control port (RS232)

The box generates outputs for the following devices:
- VAXstation keyboard port (RS423, Digital protocol)
- VAXstation mouse port (RS232, Digital protocol)
- 3 X-terminal keyboard ports (PS2, PS2 protocol)
- 3 X-terminal mouse ports (RS232, Logitech protocol)
- VAXstation control port (RS232)

An internal microprocessor accepts keyboard and pointer data from the input devices and sends it to the selected output devices, converting keyboard and pointer protocols if necessary. The VAXstation control port is used to receive screen selection commands from the VAXstation and to send knob changes to the VAXstation.

V. SECURITY

Security for access control to accelerator devices and control programs was an issue long before office and long haul consoles arrived. Increased numbers of consoles called for changes to a nearly wide open system. The strategy chosen attempts to balance protection, implementation cost, and risk, yielding a system that is accessible and accountable.

This control system encompasses and serves several control room environments. Further, setting access is required for engineering test and development, accelerator experiments, and software developers. Tools and applications have been developed to allow the Accelerator Operations group to configure the accessibility of programs and devices.

Each console is a member of one or more classes of consoles that determine what application programs may be run and devices set. Further each console has a setting lock with a variety of privileges determining its behavior. There are four ways that a user is typically denied setting access:

1. cannot run any program. The change-program lock is locked for this console. It must be unlocked by the Main Control Room.
2. cannot run the program. This console is not a member of the classes of consoles allowed to run the selected program.
3. program will run but cannot do settings because this console is not a member of the classes of consoles allowed to set with this application.
4. program will run but cannot do settings because the console's setting's lock is locked, and the user is offered no choices for unlocking, and the Main Control Room elects not to unlock this user remotely or is not able to unlock the user remotely.

When settings are performed, information about the settings are queued, then sent to a central settings logger. Application programs exist to view logged settings with a variety of presentations. An X-server console on the Main Control Room crew chief's desk is normally configured to display in real time the settings performed from consoles not in a control room environment. That number of settings is small, and coupled with the run and set control imposed on applications and devices offers the operating crew the confidence that they are in control of the accelerator.

VI. CONCLUSIONS

The new accelerator consoles are providing greater accessibility to accelerator information to users and richer tools and a more productive development environment to programmers. The growth in numbers and wide geographical dispersion of accelerator consoles present throughput and security problems that require continuous attention.

VII. REFERENCES

[1] Cahill, K.J. and Smedinghoff, J.G. (1989). Converting the Fermilab Accelerator Control Consoles to X-Window Workstations, Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems, pp. 442-445
[2] Briegel, C., Johnson G., and Winterowd, L. (1989). The Fermilab ACNET Upgrade, Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems, pp. 235-238
[3] Lucas, P.W. (1991) The Status of ACNET - Fermilab Accelerator Control Network, these proceedings

A Virtual Control Panel Configuration Tool for the X-Window System

Jeffrey O. Hill, Leo R. Dalesio and Debona M. Kerstiens
Los Lamos National laboratory, Los Alamos, NM 87545

Abstract

Computer Graphics Workstations are becoming increasingly popular for use as virtual process control and read back panels. The workstation's CRT, keyboard, and pointing device are used in concert to produce a display that is in essence a control panel, even if actual switches and gauges are not present. The code behind these displays is most often specific to one display and not reusable for any other display. Recently, programs have been written allowing many of these virtual control panel displays to be configured without writing additional code. This approach allows the initial programming effort to be reapplied to many different display instances with minimal effort. These programs often incorporate many of the features of a graphics editor, allowing a pictorial model of the process under control to be incorporated into the control panel. We have just finished writing a second generation software system of this type for use with the X-window system and the Experimental Physics and Industrial Control System (EPICS). This paper describes the primary features of our software, the framework of our design, and our observations after initial installation.

BACKGROUND

The EPICS[1] control system consists of an input-output controller subsystem that communicates via a software bus with many general purpose or application specific control system components. The input-output system provides the time critical and hardware specific portions of the control system. The software bus or "channel access" provides standardized communication between control system components over a local area network. EPICS has the following general purpose components: an alarm manager, an archive subsystem, a timing subsystem, and the operator interface which this paper describes.

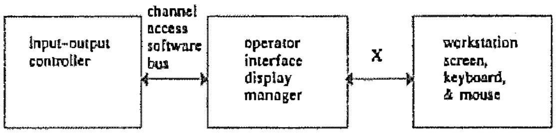
A first generation operator interface, developed for the telescope control system, has been deployed at the Argonne National Laboratory and elsewhere by the Los Alamos National Laboratory[2]. This operator interface was next ported for use on the EPICS control system and used on the Ground Test Accelerator and related test stands at LANL. This first generation operator interface proved the effectiveness of the virtual control panel technique for reducing the applications programming effort[2].

This paper describes the design and implementation of a second generation operator interface. We wanted to replace the obsolete graphics platform on which the first generation was built with the open X Window System standard. Other goals were faster display startup and update rates, and more effective configuration tools[3]. We also wanted to build a proper foundation on which we could build future extensions and enhancements, while spending less of our time on software maintenance.

FEATURES

The second generation operator interface or OPI consists of an editor which is used to create and configure virtual control panels and a display manager which activates them. A display file containing the virtual control panel description is the only form of communication between the two programs.

Once activated, displays can be used to monitor and control process variables. The display manager accomplishes this by responding to external events and updating the screen. External events include keyboard input, mouse input, or process variable state changes.



Operator interface displays are configured with a graphics editor capable of manipulating the normal complement of graphics object primitives such as rectangles, lines, ovals, arcs, and text. In addition to these primitives, the editor can also configure a full complement of process variable control and read back components such as indicators, meters, buttons, and menus. Once created, one or more graphics objects can be selected for cutting, copying, pasting, moving, or scaling. These techniques can be used to move groups of objects between several operator interface displays under edit on the same workstation. The editor also supports features for productive alignment and even distribution of object groups. The editor saves a finite list of all previous operations so that they may be individually undone at the operator's discretion.

A graphics form for modifying attributes is provided for each type of object which can be created by the editor. The forms have entries for every operator modifiable attribute even if some attribute modifications, such coordinate translations, can be carried out more efficiently with the mouse. These graphics forms or property sheets provide a simple and consistent method for entering the more complex configuration required by process control and read back components such as plots or indicators. For example, a bar indicator might require entry of a process variable name, labeling option, direction of increase, and a color modifier (the color could be static or based on an alarm condition).

---

*Work supported and funded under the Department of Defense, US Army Strategic Defense Command, under the auspices of the Department of Energy.

The operator may choose to create a private color pallette for his display or share one created for another display. Both the number of colors and the hue of each color are configurable from the editor.

Text is displayed within a bounding box. This allows text object coordinates to be scaled the same as any other object on the screen. If a group of objects containing text is scaled then the text font size is appropriately scaled also. If the exact font specified by the operator during display configuration is unavailable in the future, then a reasonable alternative font is used.

Graphics primitives may be modified for dynamic operation. For instance, the visibility or color of a rectangle may be based on the state of a process variable.

Operating displays detect loss of network connection to process variables. Any graphics objects which are attached to a lost connection are displayed in a special color with the process variable name and an explanation of the problem. Graphic objects are automatically drawn again in their correct colors once normal network communication resumes.

Macro subsitution of process variable names allows a single display description to be used for a series of replicated devices at run time. For example, when the display is configured the operator could type in 'POWER_SUPPLY_$(DEVICE_NUMBER)' for graphic object's process variable name. When the display is activated the operator could specify 'DEVICE_NUMBER = 4' and the resulting process variable name used would be 'POWER_SUPPLY_4'.

DESIGN

We chose to use an internal display list definition to describe the contents of the display files configured by the editor and executed by the display manager. This approach gave us the most flexibility to add information into the display list definition that was specific to process control applications.

Our display list can take either a binary or ASCII form. We use the binary form when the display is executed for improved performance and compact storage. ASCII display files can be safely converted to any future binary display file format. This allows us to rearrange the binary display file format in future releases of the operator interface. This feature has already proved itself valuable when we realigned the binary display file so that it would execute properly on both the SPARC and 68k processor architectures. The ASCII display list format also allows an easy way to convert between the binary display list formats of incompatible architectures. For example, one might first convert a big endian processor architecture display file to ASCII followed by conversion to a little endian processor architecture binary display file.

Additions to the display list description often require no code changes to the display configuration editor. We have a source file which describes the binary and ASCII format of the display list. This file is run through a macro processor which generates two C language include files. One of these is a binary description of the display list format in the form of C structures. The other is a description of the display list format in the form of arrays of a structure describing structure fields. It is this additional information that allows the editor to configure a new addition to the display list description without, in many cases, writing additional editor code. This design has significantly reduced the number of lines of code in the editor, thus making it easier to write and maintain.

Both the display manager and editor were carefully written to use only the parts of the host operating system that are used by the X window system libraries. This should make it an easy task to port the operator interface to any of the numerous environments that currently run the X window system. In addition, care was taken to place all external variables in allocated memory so that our code would easily port to shared memory, multitasking environments such as VxWorks. This gives us the option of creating a stand-alone system that hosts both the input-output controller and operator interface software without the need for a communications network.

Under UNIX, the display manager can run each display with a separate process or it can run many displays from one process. The latter provides for improved display startup time, because we don't have to wait for process creation to activate a display. A file descriptor manager library based on the system call select ( ) provides the display manager with the ability to respond asynchronously to events from both the X window system and channel access simultaneously.

The display manager does not queue up each update from channel access prior to making a graphics update on the screen. If so the display manager would grow further and further behind if updates were generated more rapidly that they could be consumed in the form of graphics updates. Instead, the display manager writes each update into memory and sets a modified data tag. When all updates have been dispatched as above then a list is scanned for modified data and the screen is updated. This method allows display manager to react asynchronously when the updates are infrequent and gracefully degrade to synchronous operation when the updates are continuous.

The display manager and editor both share a common custom graphical user interface tool kit written specifically for process control applications. When the project began the XView and Motif tool kits were unavailable. The project athena and HP widget sets were available at the time but did not provide the features we needed. Attempts to use the Xt subclassing techniques to extend the functionality of these tool kits still did not provide the features we needed.

Another factor contributing to our choice of tool kits was our decision not to create virtual windows with the X libraries for each of our output-only graphics devices such as meters, bars, and indicators. This choice improves start up time and decreases overhead since some of our displays have a large number of these devices. Keyboard and mouse input events generated under the X window system are associated with the virtual window that generated them. The X window system also provides a shifting type coordinate translation for all drawing operations performed within a virtual window. Both of the above are the normal benefits of using virtual windows. However, our output-only graphics devices do not receive mouse or keyboard input. Similarly, we must already perform a scaling type coordinate translation so that graphics devices will appear on the display with the proper height, and width as specified by the display list. All X tool kits investigated create a virtual window for each object.

Each of our user input devices such as menus, keyboard input, valuators, and bottons does have an X library generated virtual window associated with it. We use the hash table routines built into the X libraries to translate window identifications provided by X into a pointer to our structures containing the graphics input device configuration and state information.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI06

CONCLUSION

We have been using our new operator interface in some installations for over a year now. The ASCII version of the display list has allowed us to continually add new features without sacrificing upward compatibility.

In both the first and second generation each display could start up several others. We prefer to set up a hierarchy where each time a display initiates another we see an increasingly detailed view of the process. The second generation operator interface's order of magnitude improvement in display start up time has made this organization practical. Likewise, faster start up times allow us to draw many reusable small displays instead of one display with a large portion of the process on it.

Our experience with the X window system has been positive. Finally, we can write graphics code that does not become obsolete with the graphics hardware. The client private color table and scaleable fonts that appear to be in release five of X11 would have saved us effort if they had been available at the start of this project.

Graphical user interfaces are arguably a revolution in computer accessibility. However, the time spent writing graphical user interfaces is a large and growing percentage of the total time devoted to a project. We have tried to reduce this percentage by providing a software tool that allows process control based on graphical user interfaces to be configured without writing any new source code.

References and Comment

1. Our control system has been renamed from LAACS to The Experimental Physics and Industrial Control System, or EPICS, since the last ICALPECS to better reflect the collaborative effort now in place.
2. Richard Rothrock, "A Workstation Operator Interface for Accelerator Control Systems: A Method for Separating Graphics Code from Control Functionality", Europhysics Conference on Control Systems for Experimental Physics, Sept. 28 – Oct. 2, 1987, Villars sur Ollon, Switzerland.
3. L. R. Dalesio, "The Ground Test Accelerator Operator Interface: The Second Generation", International Conference on Accelerator and Large Experimental Physics Control Systems, October 30 – November 3, 1989, Vancouver, Canada.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI07

X-Window for Process Control
in a
Mixed Hardware Environment

Matthias Clausen, Kay Rehlich
Deutsches Elektronen Synchrotron
D-2000 Hamburg 52

Abstract

X-Window is a common standard for display purposes on the current workstations. The possibility to create more than one window on a single screen enables the operators to gain more information about the process. Multiple windows from different control systems using mixed hardware is one of the problems this paper will describe. The experience shows that X-Window is a standard per definition, but not in any case. But it is an excellent tool to separate data-acquisition and display from each other over long distances using different types of hardware and software for communications and display. Our experience with X-Window displays for the cryogenic control system and the vacuum control system at HERA on DEC and SUN hardware will be described.

I. Cryogenic Controls

A. Components

The cryogenic control system for the HERA collider -which has a circumference of 6.3 km- is based on a commercial, distributed control system called D/3. The backbone of the system is a redundant communication link using HDLC protocol and a token passing algorithm(Fig. 1). Display and control functionality a separated from each other in the individual display-(DCM) and process control computers (PCM). The cryogenic processes, as there are: compressors, coldboxes, helium distribution in the HERA tunnel to the 422 superconducting dipole- and 224 quadrupole magnets, low temperature measurement, superconducting cavity control, supervisory control for the ZEUS solenoid, controls for the magnet test hall etc. are controlled by means of the PCMs. More than 3000 analog and 3100 digital signals are scanned,archived and calculated in control loops and logic devices. The scantime for the individual points is defined to be between 0.25 [sec] for fast control loops and 3 [sec] for temperature read-out of the HERA magnets. All points are checked for over and underrange, and high and low limits. Alarms are send to various printers throughout the system according to the alarm destination index (ADI). The printers can be host based or connected to terminal servers.

B. Access

All process points can be accessed from any PCM and DCM in the whole system. This way no 'special' consoles exist in the system for process control. There are some consoles that have additional/other functionalities like annunciator panels with function keys and X-Window displays with or without the full access to the process. This (X-)extension to the existing system is very useful since the architecture of the D/3 system does only foresee consoles directly connected to one of the DCMs. X-Displays of the cryogenic control system are now running in various places at DESY: In the main control room where the D/3 link is not jet installed, and where ever it is useful for the

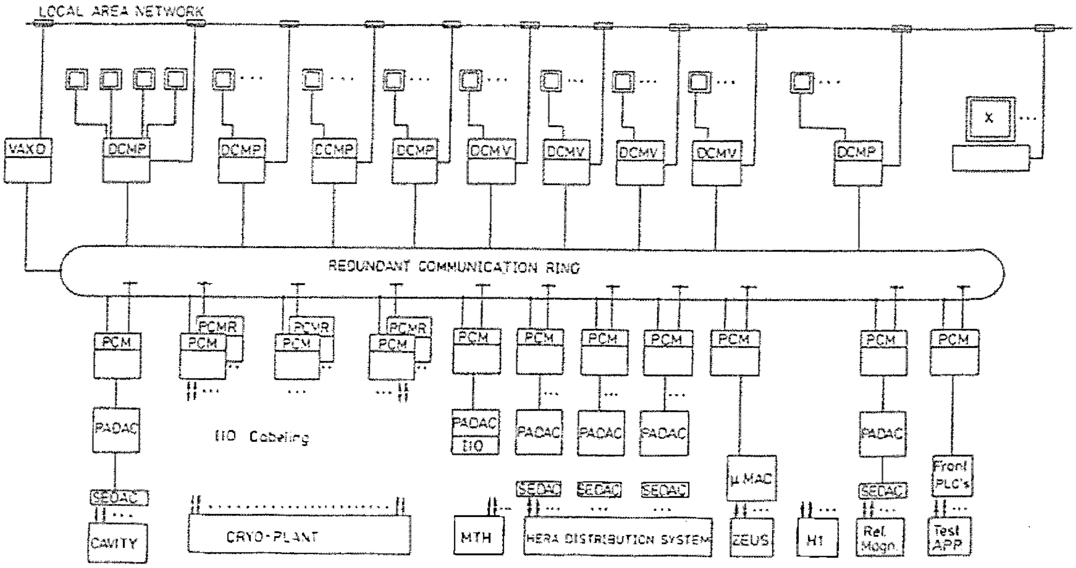


Figure 1. The Cryogenic Control System

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI07

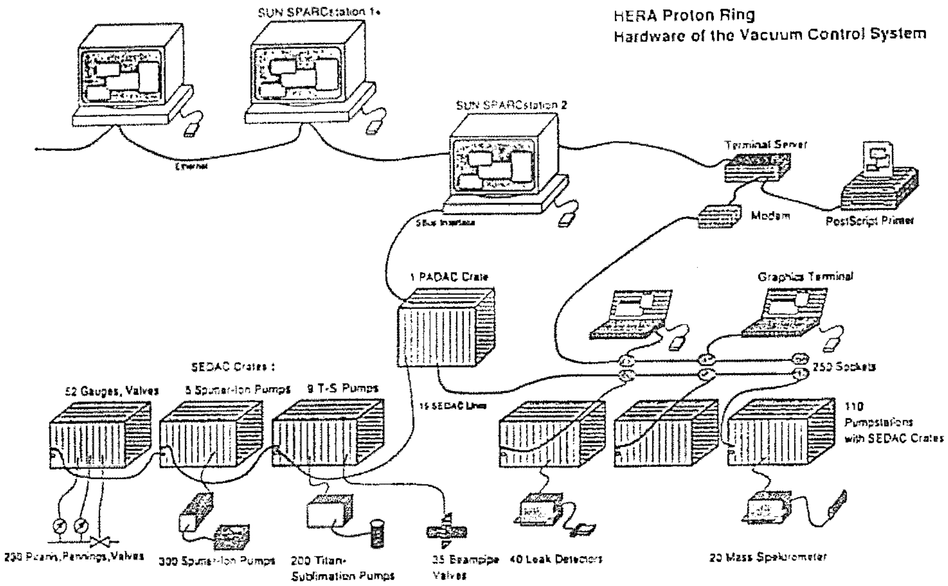


Figure 2. The Vacuum Control System

vacuum operators and in smaller control rooms like the magnet testhall and at the ZEUS experiment. The X-Clients for these displays run on one of the VAX-based DCM's (DCMV) in the cryogenic controls cluster.

II. The Vacuum Control System

A. Components

The insulating vacuum of the superconducting magnets, the beam pipe of the ring and the insulating vacuum of the helium distribution line are part of the vacuum control system. We have installed 700 pumps of different type, 1200 gauges and 360 valves. These devices are controlled by several Sun SPARCstations and graphic terminals. The workstations are linked by an Ethernet network running the TCP/IP protocol. One workstation has a hardware connection to all vacuum devices and runs the server program.

B. Communication

The device server program reads out the equipment every four seconds, checks the status and acts on special situations, writes out error messages, stores measurement and status readings in a database and writes changes of the values to a history file on disk. All other programs can access the device server by remote procedure calls (RPC) to allow calls from different

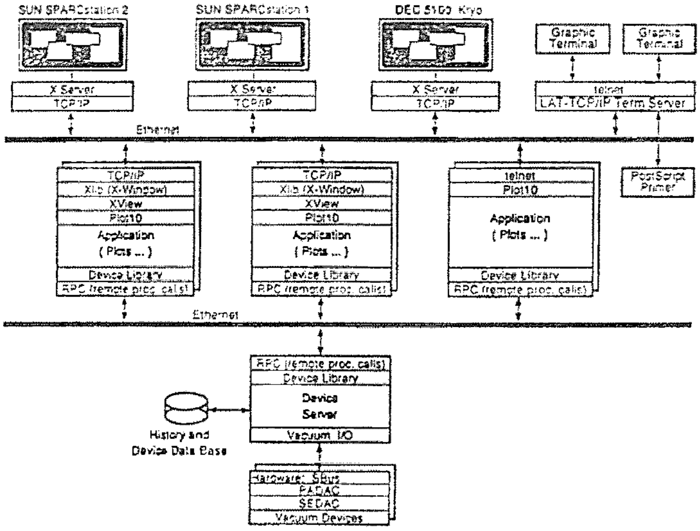


Figure 3. Vacuum Communication

workstations on the network. The data transfer of the RPCs is done in a machine independent manner by using external data representation (XDR) protocol. This gives all sorts of computers of different manufactures access to the vacuum database by means of converting the data to a standard format on the network.

All application programs are requesting device related information by the specification of a device type, a device position, the type of function to perform and the property to transfer. The server responds with an error code and a single data structure or a data block with actual measurements of several similar devices or the history of one device.

C. Data Presentation

The data are presented in a graphical way to the user, either in forms of diagrams or in plots with time or position dependence. Presentation devices are: workstations running Open Look; Tektronix compatible terminals for use in situations where hand held terminals -designed by DESY- have to be used and matrix color printer or postscript printers.

The transport protocol for the workstations is X11 (X-Window) to allow displays on different types of machines. The next layer on top of the Xlib routines of X-Window is the XView library, an interface to Open Look graphics, which provides a simple access to X11. A further layer is the Plot10 graphic standard. This additional interface gives the ability to display graphics on Tektronix type terminals as well as X11 windows from one source.

All user interaction is done by mouse input. The operator can scale plots and diagrams, switch vacuum devices or display windows with history plots or faceplates by a simple mouse click. A window displaying the timedependence of the pressure is for example generated by a click on the gauge reading in a diagram. Additional services, like printing or refresh rates etc., are provided by pull down menus.

III. Reliability

A. Long term operation

Compared with a host based display, the X-display has to rely on the availability of many components like cpu's networks and software. Fortunately the development of soft- and hardware has made a great progress during the last years. We are running X-displays from different machines to one X-Server over the Ethernet without major problems.

B. Display Activation

Calling up a host based application is as easy as for any other application. But calling an X-Display and redirecting it to another machine is not easy to perform. Specially operators on a control room should not have to deal with this task. Tools are needed to make life easier. Under DECnet you can call DECnet objects installed on the application server. Starting up the image and redirecting the output to the calling machine is done automatically.

Display call-up:

```
DEC: $ set display/create /node=HOST    /transport=
[(DECnet), TCPIP, LAT, LOCAL]
$ run application
SUN:  : application -display IP-HOST:0
```

C. Redundancy

Even if the operator does not have to deal with the proper X-setup, he will be in trouble if the application server with the X-client goes down. If possible there is a chance to solve even this problem. If the application can be run on a DEC LAVC(local area VAX cluster) the X-Client can be started as a Batch job. Defining the queue as generic with queues on other LAVC members the task will be automatically passed to another machine in the cluster and send the same output to the X-Server. The operator just observes a short blackout before he can continue work. This is a big advantage compared with host based applications.

D. Long Distance

End of 1990 a demonstration system was installed on an exhibition at CERN. The X-Window communication had to pass a well saturated 64 kbit line between DESY and CERN. The demo was running over a long time period with reasonable results. The most important experience was, that the X-System does not suffer from long answering times. The display was slow but stable.

IV. Fonts

A. Standards

Even though the X-System is a standard, this does not really mean that you will be able to run any application on any X-Window terminal. The X-Window system disposes several standard fonts. If you keep using these fonts you will normally

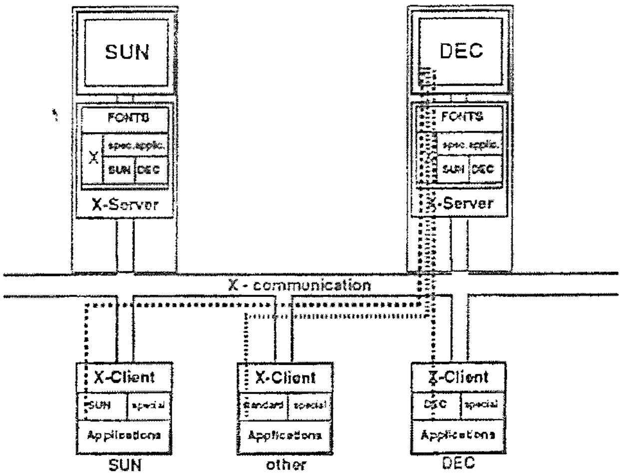


Figure 4. Standard X-Window Fonts

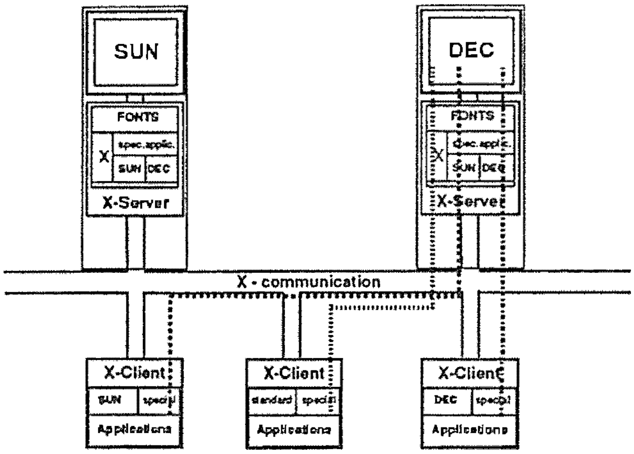


Figure 5. Special Application X-Window Fonts

be able to run this application on any X-Server. Standard fonts which are not known will be displayed in another font -the one that fits best-. There are ways to write non standard applications if your program explicitly needs a special set of fonts. In this case the font will not be replaced by other fonts, but you will have to load the fonts to your server. Not enough, you will also have to put them into the right directories and make them known to the system.

B. Different Systems

If you have to include fonts from other systems you will find out that there is no standard for the file format of the fonts. Fortunately there is a way to exchange the font information. This is done in the Binary Distributed Format (BDF). The files are ASCII files readable by any system. These files have to be compiled with a font compiler which is special for every system.

Font compiler:
DEC: $ DECW$FONTCOMPILER
SUN: : convertfont

Afterwards the fonts must be included in the standard font directories or the actual directory must be included in the font path.

Font path:
DEC: $ sys$manager:DECW$PRIVATE_SERVER_SETUP
SUN: : bldfamily

IV. Networking

A. Communication Protocols

The X-System is designed to be network and vendor independent. The most common protocols on top of which the X-protocol runs are: TCPIP,DECnet,LAT local. TCP/IP is the common protocol for UNIX systems. Also for DEC machines several implementations of this protocol exist. DECnet is a proprietary protocol of Digital Equipment. There are several implementations on other systems i.e. SUN-OS. LAT is a protocol from DEC designed for LAN transport only.

B. Network Load

Several measurements between different systems using different protocols have shown that the applications we use for the cryogenic and the vacuum controls need very rare band width of the Ethernet. A graphic with about one hundred bargraphs updated each second takes less than 0.4% bandwidth. In this case it was X-Window over DECnet. Using 'pure' DECnet to transfer the graphic information took about 0.2%. There is an overhead for the X-display but the real numbers can not be drawn from this measurement since the numbers are at the low end of the resolution of the network monitor we used. All the relevant informations for the display update must be packed into less than 8 DECnet packets. The same results were measured for the X-display using TCPIP as the transport protocol between a SUN and a DEC machine. If client and server are both connected to the same LAN it seems to be reasonable to use LAT as a transport protocol since LAT uses smaller packages. Since the measurements with LAT were made with a DEC-X-Terminal which obviously was equipped with a to small CPU, measurements with other X-Terminals should be made. The only measurable network load occurs during the initialization of the display on the X-Server. But also this takes only some percent of the bandwidth.

V. Conclusions

Almost one year of operating experience showed a reliable operation of the vacuum control system. The provision of two network layers between the vacuum equipment and the user display gives much of freedom in transferring data and graphics to different sorts of computers. It makes good software development capabilities available by allowing application programs to be tested with real data without interrupting the vacuum process. In general the overhead in speed of the network is negligible. Only some data bound applications are performing better by sending graphics data instead of fetching data from the history memory across the network. The integration of vacuum displays on the cryogenic workstation -and vice versa have proved to be very useful for the operators and will be expanded in the future.

VI. References

[1] G. Horlitz, "Refrigeration of a 6.4-km Cirumference, 4.5 Tesla Superconducting Magnet Ring System for the Electron Proton Collider HERA", Proc. 10th Intl. Cryo. Eng. Conf., Butterworth, Guilford, UK (1984), p.377.

[2] M. Clausen,"Experience with a Process Control System for large Scale Cryogenic Systems, Advances in Cryogenic Engineering, Vol. 33, Plenum Press, New York and London

An Open Software System Based on X Windows for Process Control and Equipment Monitoring

A. Aimar, E. Carlier, V. Mertens
European Organization for Nuclear Research (CERN) - SL Division,
CH - 1211 Geneva 23, Switzerland

Abstract

The construction and application of a configurable open software system for process control and equipment monitoring can speed up and simplify the development and maintenance of equipment specific software as compared to individual solutions. The present paper reports the status of such an approach for the distributed control systems of SPS and LEP beam transfer components, based on X Windows and the OSF/Motif tool kit and applying data modeling and software engineering methods.

I. INTRODUCTION AND MOTIVATION

A. Equipment

Equipment for SPS and LEP beam transfer at CERN comprises systems like the SPS injections, extractions, targets, dumps, and collimators, and the LEP injections and separators. In total some 80 distinct systems spread over both accelerators and the fixed-target areas have to be controlled. At present, new control systems for the LEP beam dump and the LEP Pretzel separators are being prepared.

Although the functionality and the composition varies considerably between the different systems, all can essentially be characterized as 'slow controls': Reaction times as seen from the main control room are of the order of seconds; any fast responses, e.g. for beam dumping, are supported by special hardware. The amount of data exchanged between the main control room and the devices is small.

B. Equipment Software

However, comprehensive equipment specific software has to be provided to achieve the desired level of abstraction towards the main control room, to allow monitoring of the equipment performance, and to dispose of efficient tools for local and remote fault-finding to help keeping down-times low, in particular in view of the volume and the distances involved.

For this sake a lot of code has been written up to now, especially with the large-scale use of distributed processing. Due to the limited manpower there is a strong risk of bottlenecks in the treatment of requests for modifications or extensions which might arise from an evolving environment or an increased sophistication of use.

C. Tool Kit Approach

This experience has encouraged us to try a different approach by replacing our equipment specific programs by a general software system or tool kit which receives its individual functionality through a formal description of the equipment and the desired function in tables. If this idea is pursued rigorously almost full separation between code and data can be obtained, leaving only pieces of specific code behind which are uneconomic to parametrize.

As possible advantages we see, besides others, a more uniform appearance of the equipment, a more transparent specification phase with improved communication between hardware and software specialists, leaving less room for misunderstandings, and a shorter reaction time for developments and modifications.

Before starting the development we made some investigations in the commercial sector. At that time we came to the conclusion that a separate development could well be justified in view of the potential problems encountered when embedding and maintaining a commercial process control system in a given and evolving environment, disregarding any price argument. With time passing by, we might however come to a different finding.

To make our task more feasible we did not attempt to write a full package from scratch but rather tried to re-use a maximum of existing packages, tools, and mechanisms, combining them into the desired product.

One of the key goals was to arrive at a portable, thus platform independent, and evolvable software system which should be easy to adapt to changing environments or increasing needs. This becomes particularly attractive in combination with the X Windows system and the OSF/Motif tool kit since they allow to perform input or visualize complex results on a large variety of media without much adaptation work.

In the following we will first give an overview of the tool kit concept and describe its key ingredients, then the chosen implementation. Afterwards, we will present results obtained in a prototype application, followed by a status report and an outlook.

II. TOOL KIT CONCEPT

The layout of the tool kit and the way the user interacts with it is sketched in figure 1.

A. General Structure

The tool kit is built up in a modular way. Each module apports a well defined and limited function which contributes to a smooth progression and testing.

Its core module is a shared run-time data base (SDB), containing all data which are system-wide accessible, surrounded by a number of peripheral modules, some of them containing their own data bases (DB) necessary to perform the required function. A server (SDBserver) provides all modules with access to the shared data base thus permitting the inter-module communication. Besides that, there is a tool for accessing the different module data bases to configure the desired application and a graphics editor and animation system to link elements in the shared data base to screen objects. The most important items will be discussed below.

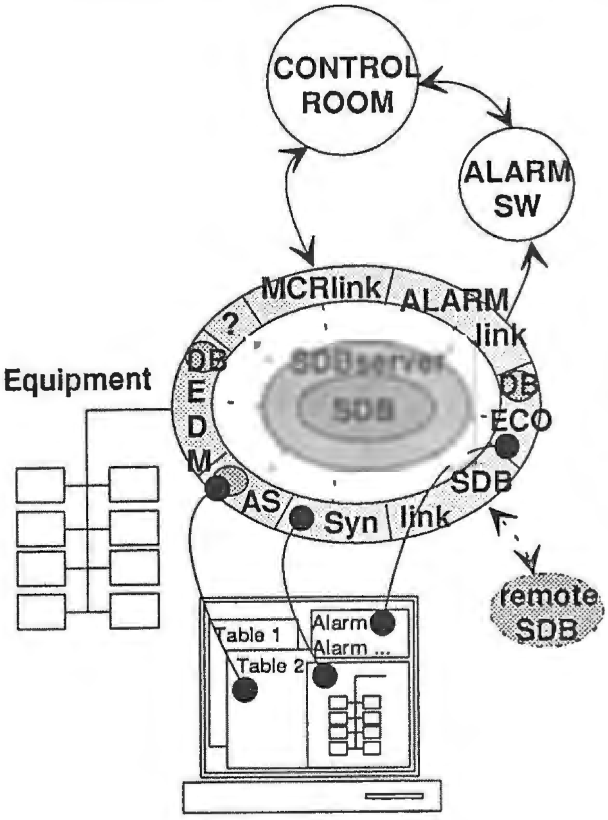


Fig. 1: The tool kit architecture and the user interaction

B. The Shared Data Base

The shared data base contains all data exchanged with the outside world at run-time, e.g. an image of all equipment channels, like voltages or temperatures, but also more abstract data like the last requested state of the equipment. Abstracting external data in terms of these values allows the system to be based on a unique and unduplicated representation of the context and to build tools for displaying, changing, and checking these values in a way completely independent from their external format.

C. The Shared Data Base Server

The use of a server for the book-keeping of shared data allows concurrent access and distributed computation in a straightforward manner. The interface to its services has been designed in terms of Remote Procedure Calls so that every client module can access the global data in a transparent and location independent way, preserving the important issues of openness and developmental possibilities.

D. The Equipment Data Manager, the Equipment Control Operation Module, and the Action Sequencer

The Equipment Data Manager (EDM), the Equipment Control Operation (ECO) module, and the Action Sequencer (AS) implement the more proper control functions.

The EDM keeps the shared data base consistent with the inputs and outputs of the attached equipment and takes care of the adaptation to special communication protocols. Special procedures like initialization routines are also located here.

The ECO checks for the occurrence of fault conditions and treats special events.

The AS takes care of the required sequences, i.e. all chains of actions which have to be performed upon a certain event or a requested state transition.

All modules are completely driven from their own configuration and run-time data bases. These contain, in case of the EDM, the complete description of the equipment, e.g. module addresses and transformation factors between electrical and physical values, or, in case of the ECO, alarms conditions, messages, and emergency actions.

E. The Shared Data Base Link Module

The definition of data which have also to be accessible from modules at remote sites is made through the Shared Data Base Link (SDBlink) module, which is in fact nothing else than a client to all involved data base servers and which has its proper data base containing the correspondences.

F. Special Interface Modules

Particular attention has to be paid to a clean functional implantation into the global accelerator control system [1] to allow a correct remote control and propagation of alarms.

For this sake two interface modules are foreseen. The first one (MCRlink; already exisiting) permits to access the shared data base through the RPC mechanism widely in use at CERN. This module will, in the future, also follow and implement the new control protocol guidelines as laid down in [2].

The second one (ALARMlink) is an interface to the general alarm system. It transforms the alarms resulting from the ECO checks into the required format and feeds it into the existing software chain.

G. User Access Tool

An adequate tool is required to access the special module data bases, i.e. the tables which allow to tailor the application, in a proper and comfortable way. Several solutions are being considered at the moment, e.g. [3].

H. Synoptics

A tool permitting to generate synoptic images and animate them in correlation with the shared data base contents and changes (Syn), based on a commercial user interface builder, working with X Windows and OSF/Motif, and producing UIL (User Interface Language) or C code, is currently under investigation.

III. TOOL KIT IMPLEMENTATION

A. The Target Environment

The environment in which the tool kit should finally run is the process controller proposed as standard interface between the equipment and the accelerator network [1], in our case, a diskless industrial PC running under the real time operating system LynxOS, together with X Windows and OSF/Motif.

B. The Development Environment

To simulate the target environment we used a diskless DECstation 5000/125, with a remote workstation acting as file server. The use of a multi-windowing environment and the fact that all important tools were already available permitted to advance quite rapidly.

To simulate attached equipment electronics we used a) a HV power supply and a programmable timing unit built up in G-64, each controlled by a microprocessor and connected to the workstation through a terminal server by RS-232 links, and b) several commercial I/O modules on a Bitbus, linked to the workstation via a SCSI-Bitbus gateway [4]. No additional driver software was needed.

C. The Data Base

In an application like this where the reliable access to a lot of data is essential, the selection of a good data base system becomes a key issue. Our investigations for a non-commercial product which has proven its quality in similar applications has led us to the choice of ADAMO [5, 6].

ADAMO (= Aleph DAta MOdel) has been developed within the ALEPH collaboration at LEP where it is now used in the fields of data acquisition, detector description, event reconstruction, and data analysis [7]. More recently other experiments have started to use it, among those the ZEUS experiment at HERA [8, 9].

ADAMO is based on the entity-relationship (E-R) model of Chen [10]. This model adopts the view that the world consists of entities - objects that can be distinctly identified and have a fixed number of characteristics called attributes - and relationships, which are associations between entities. A data model [11], generally spoken, is a strategy for data organization which includes formally defined data structures, operators to act upon the data, and validation procedures to ensure that the data obey the imposed constraints.

The ADAMO system provides all these features in a form suitable for scientific computing where numerical algorithms are important. The choice of ADAMO is also supported through its proven portability to the major hardware platforms and operating systems used at CERN (in fact, it is based on the CERN software libraries).

D. CASE Support

CASE (Computer Aided Software Engineering) tools provide methods for describing graphically and consistently a software system in its life cycle through analysis, design, coding, and maintenance [12] thus speeding up the development.

Such a tool, called StP (= Software through Pictures, from Interactive Development Environment), has been employed for the construction of the tool kit, using an ADAMO-Interface to the Picture Editor of StP. This allows, starting from the design of the data bases in E-R diagrams, to generate automatically the data structures in a format that can be accessed and manipulated from the desired programming language through a run time library.

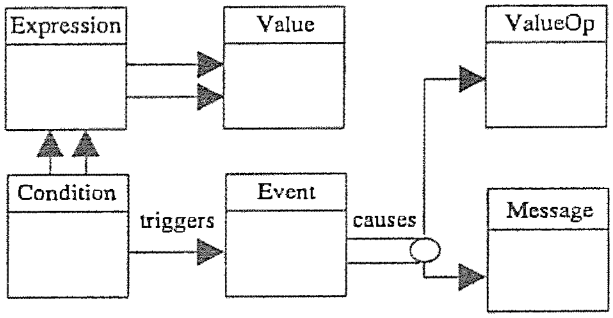


Fig. 2: E-R diagram of the ECO module (simplified partial view)

As an example of the applied technique, figure 2 shows a simplified part of the E-R representation of the ECO module which can be interpreted as follows: A 'Condition' on which the system should check is made up from 'Expressions' with 'Values' as operands. Such a 'Condition', when fulfilled, can 'trigger' an 'Event' which 'causes' either 'Messages' or 'ValueOperations'.

E. Use of Standard Tools

To provide a maximum of portability, standard tools have been employed wherever possible.

The tool kit itself has been written in C. Since the UNIX world is our main target the internal communication layer through Remote Procedure Calls has been implemented using NCS (Network Computing System by Apollo Inc.) [13].

The windowing environment for all interfaces used for customizing or output is provided by X Windows and OSF/Motif.

IV. EXPERIENCE AND RESULTS

A prototype application has been realized which comprises about 30 digital and analogue I/O channels. On the DECstation 5000/125, the CPU time needed to perform a full cycle of readings and checks is currently about 45 ms (of this, about 30 ms are consumed by the data acquisition and mostly due to the way the equipment has been attached). The application performs very stably.

The allocation of virtual memory is around 6 MB. To have a safety margin when running in a diskless environment together with other processes a total memory size of 16 MB seems adequate.

V. STATUS AND OUTLOOK

The following elements of the tool kit have been commissioned until now: The shared data base, the data base server, equipment managers for Bitbus and multiple RS-232 links, the alarm module, and the RPC interface.

The most prominent future task is to transfer the whole system onto the desired target environment, a PC running under LynxOS.

Once this has been accomplished, we first plan to streamline the product in terms of simpler interaction (table entry and user interface), to add protection mechanisms against unauthorized use, and to take care of the display of actual trends, data logging, and re-play of stored data. One of the more long term goals is to provide a higher degree of support for fault finding (on-line help) through the application of expert system techniques.

The first test application in a larger project is planned for spring 92.

VI. CONCLUSION

The extensive use of existing tools, like the data management system ADAMO or the CASE tool StP has made possible to optimize the cycle between analysis, design, coding, and implementation and to arrive, in a relatively short period and with little manpower effort, at a satisfactorily working prototype system.

The use of recognized standards for the underlying mechanisms makes this tool kit a highly portable product which could find its application also in other areas where slow control systems are needed.

VII. ACKNOWLEDGEMENTS

The authors would like to thank the members of the CERN-SL/CO group for their support and for providing the necessary hardware and software infrastructure.

It is a pleasure to acknowledge the support received from the DEC Joint Project Office at CERN.

We are indebted to J. H. Dieperink, S. Fisher, P. Palazzi, and W. von Rüden for their contributions and A. Lovell for his help.

We would like to thank the organizers of this conference for their efforts.

VIII. REFERENCES

[1] R. Rausch and Ch. Serre, "Common Control System for the CERN PS and SL Accelerators", this conference.
[2] WOPRO Working Group, reported by G. Benincasa, "Control Protocol: The Proposed New CERN Standard Access Procedure to Accelerator Equipment: Status Report", this conference.
[3] A. Aimar, "VERDI, Visual Entity Relationship Data Interface", thesis, Department of Computer Science, Torino, Italy, 1989.
[4] A. Aimar et al., "SPS/LEP Beam Transfer Equipment Control Using Industrial Automation Components", this conference.
[5] M. G. Green," The ADAMO Data System, An Introduction for Particle Physicists", Royal Holloway and Bedford New College, UK, RHBNC 89-01 and CERN/DD/US/131.
[6] S. M. Fisher and P. Palazzi, "Using a Data Model from Software Design to Data Analysis: What Have We Learned ?", Comp. Phys. Comm., 57, pp. 169 - 175, 1989.
[7] Z. Qian et al., "Use of The ADAMO Data Management System Within ALEPH", Comp. Phys. Comm., 45, pp. 283 - 298, 1987.
[8] H. Kowalski et al., "Investigation of the ADAMO Performance in the ZEUS Calorimeter Reconstruction Program", Comp. Phys. Comm., 57, pp. 222 - 224, 1989.
[9] R. Loveless, "ZEUS Hardware Control System", Comp. Phys. Comm., 57, pp. 313 - 315, 1987.
[10] P. P. Chen, "The Entity-Relationship Model - Toward a Unified View of Data", ACM Trans. Database Systems, 1, pp. 9 - 36, 1976.
[11] D. C. Tsichritzis and F. H. Lochovsky, *Data Models*, New Jersey: Prentice Hall, Englewood Cliffs, 1982.
[12] D. J. Hatley and I. A. Pirbhai, *Strategies for Real-time System Specification*, New York: Dorset House Publishing, 1987.
[13] M. Kong et al., *Network .Computing System Ref. Man.* New Jersey: Prentice Hall, Englewood Cliffs, 1987.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI09

Porting Linac Application Programs to a Windowing Environment

J.M. Nonglaton, U. Raich
CERN 1211 Geneva 23 - CH

Abstract

We report our experience in porting Linac application programs written for Camac controlled hardware consoles to an X-Windows/Motif based workstation environment. Application programs acquire their parameter values from a front end computer (FEC), controlling the acceleration process, via a local area network. The timing for data acquisition and control is determined by the particle source timing.

Two server programs on the FEC for repetitive acquisition and command-response mode will be described.

The application programs on the workstations access a common parameter access server who establishes the necessary connection to the parameters on the FEC. It displays the parameter's current values and allows control through Motif widgets.

An interactive synoptics editor and its corresponding driver program allow easy generation of synoptics displays and interaction through command panels.

I. THE EXISTING SYSTEM.

The control system for our Proton Linac has been designed and implemented in the mid seventies. The system is based on a single PDP-11 minicomputer running the RSX-11 operating system. Because of the memory sizes available at that time all system software and a major part of the application programs has been written in Macro-11.

The software system consists of 3 logical parts:

- The equipment driving software: all equipment is interfaced through serial Camac. This software part contains a "central request processor" collecting all requests for Camac access and sending out Camac command in synchronism with the particle source timing.
- Software managing the operator consoles the consoles are interfaced through parallel Camac
- The application programs.

For historical reasons the Linac control system is the only accelerator control system in our division that uses PDP-11 computers. All other machines are controlled with Norsk Data equipment. The consequence of this is the impossibility to access the Linac control system from the general purpose operator consoles in the main control room (MCR), since the computer networks of the two types of systems are incompatible. The first goal for this project was therefore to give access to Linac parameters through the new workstation operator consoles.

II. THE NEW SYSTEM

The old PS control system is in a process of rejuvenation [1] according to the new common architecture for CERN accelerator [3]. In this global plan a special plan was defined for our Proton Linac. This was especially needed by the impossibility to maintain any more the equipment of the Linac consoles. It was also an opportunity to gain experience in windowing environment and in porting old style application into this environment. To achieve this halfway solution, we decided to connect the PDP11 front end computer to Ethernet network and to use Decnet communication package between these front end and the Ultrix workstation. This network software was the only one supported by the manufacturer DEC on the RSX11-M operating system of the PDP11. We therefore needed to write Decnet server for PDP11 to allow remote access from the workstation to the equipment.

III. NEW SOFTWARE WRITTEN FOR THE PDP-11

Because the Proton Linac is almost permanently running during the whole year, only a gradual switchover to the new system seems possible. We therefore decided to rewrite the major application programs under X-Windows/Motif for the workstations and leave the change of the parameter access processes to the DSC for a later date. In order to be able to operate the Linac from a workstation we identified the following software as absolutely indispensable:

- access to any single parameter for acquisition and control;
- synoptics;
- logs;
- several application programs, especially beam diagnostics.

Leaving the old consoles in the Linac control room in place and the old programs accessible, it was possible to install the new application software without interference in the operation of the accelerator. To make the parameter-access-part of the PDP-11 system available to the external world, two "server programs" on the PDP-11 had to be developed. The first one (VXS) operates in command-response mode, waiting for a command sent to it over the network. This command is translated into calls to the hardware driving software and submitted. The response is put back into a network packet and sent back to the requester. The command-response server accepts commands to acquire equipment data and status, to control equipment and to get database information like min/max values, convertion factors and the like.

The second server (SNS) gets a collection of equipment parameter names and performs continuous acquisition on these parameters. The acquisition values are periodically sent back to the requesting client program. The timing for the acquisition is given by the proton source timing which pulses at a rate of 1 Hertz.

The only network software available under RSX11-M is an incomplete implementation of DecNet, which does not allow to open more than 1 logical link to another task. Real server processes are therefore impossible. The only way to allow several tasks access to Linac parameters is to duplicate the "servers".

In addition to the "servers" described above (SNS,VXS), which are the most important ones, a GPIB server, a CAMAC access server and a server for starting any PDP-11 application have been written.

IV. NEW SOFTWARE ON THE WORKSTATIONS

A. The knob server

In order to harmonize the user interface for parameter access of all applications programs a "knob server" has been developed on the workstation. A typical application program will register all parameters it needs to be updated regularly on SNS and allow the user to select a limited number of parameters for control. These parameters are then submitted to the knob server, who establishes the DecNet connection to the "server" programs on the PDP, pops up a parameter access and control panel on the workstation screen and manages the command and data transfers to and from the servers.

Figure 1 shows the interconnections between the application programs on the workstations, the newly written "servers" on the PDP-11 and the old parameter access software on the PDP.

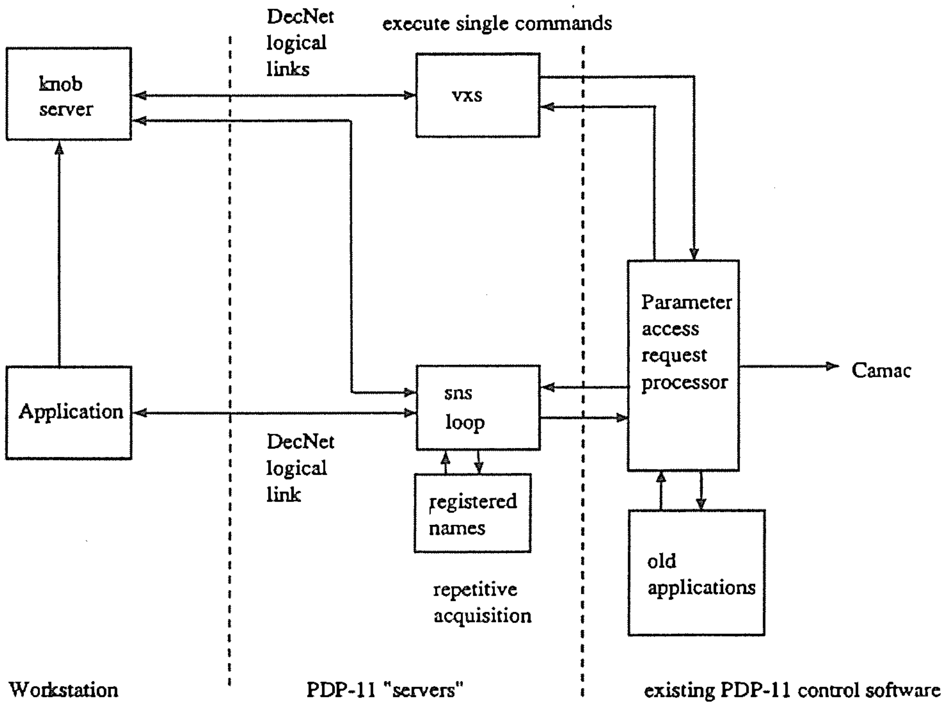


Figure 1 : The system constituents and their interconnection

B. Single parameter access

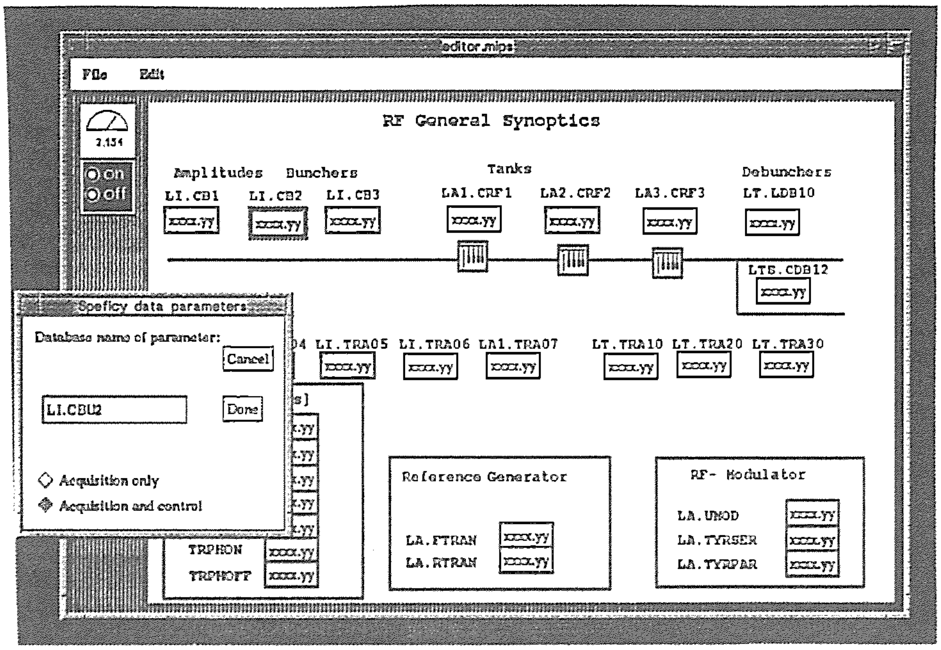
In order to get access to any single parameter a touch panel simulator has been developed. Each parameter is assigned a button which can be reached over a touch panel tree. Activation of a button may have one of three possible effects:

1) display of a new page;
2) activation of an application program;
3) connect to a parameter through the knob server.

This program has been written as a temporary tool to allow parameter connection and starting of applications until a more performant generic tool of this type (the console manager) will be available.

C. Synoptics

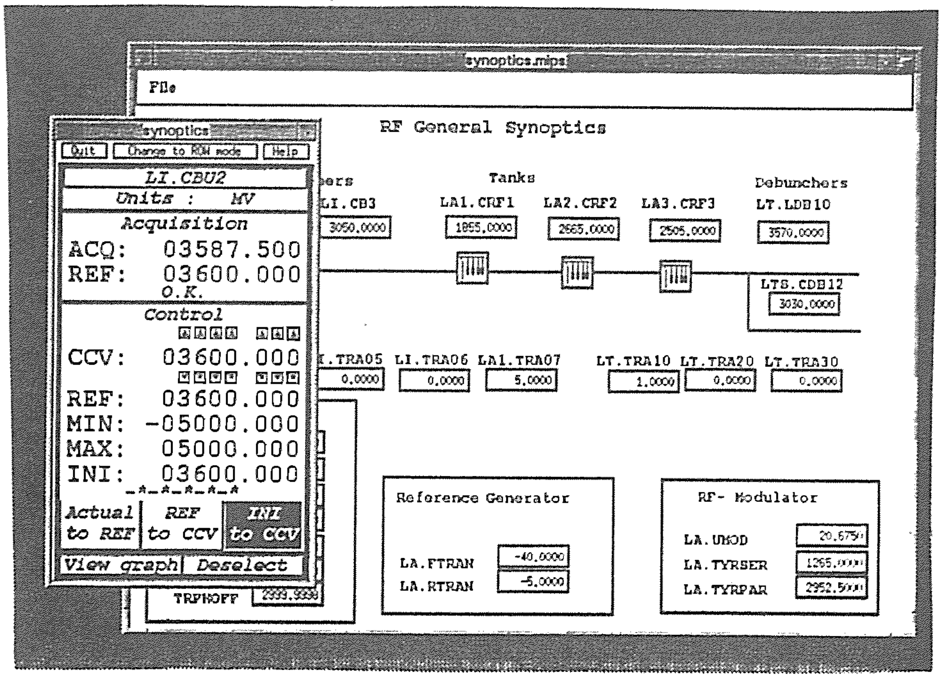
Many application programs simply show the status of an accelerator subsystem like a beamline, an RF subsystem, a timing subsystem, etc. A static background picture shows a representation of the subsystem and active boxes indicate the equipment state with different colors or give their acquisition values in form of numbers. Those types of applications can be generated completely interactively using a synoptics editing tool and its corresponding runtime part. The editor reads a previously designed bitmap (paint programs are available under Motif) and allows to place active boxes freely on top of the background picture. The editor allows to select any of the boxes and connect them to an equipment parameter by specification of the parameter name, and properties like acquisition only or acquisition and control. A status box contains a symbol representing the connected parameter and a color corresponding to the parameters state. For status boxes we can therefore specify the color for a combination of status bits and the filename of a bitmap containing the symbol icon. The result of the editing session is saved on a synoptics specification file which is consecutively read by the runtime part of the synoptics package. The runtime part will reconstruct the image, bringing up the background picture and all boxes at the predefined places. Knowing the parameter names from the output file of the editor the boxes are activated by making the necessary calls to the equipment access routines. Figure 2 shows an editing session for a synoptics application while figure 3 shows the running synoptics where a parameters has been selected for control. The control panel displayed by the knob server can be seen.



Database name of parameter: LI.CBU2

◇ Acquisition only
◈ Acquisition and control

Figure 2. A synoptics editing session



LI.CBU2
Units : MV
Acquisition
ACQ: 03587.500
REF: 03600.000
O.K.
Control
CCV: 03600.000
REF: 03600.000
MIN: -05000.000
MAX: 05000.000
INI: 03600.000

LI.CB3 3050.0000
LA1.CRF1 1855.0000
LA2.CRF2 2665.0000
LA3.CRF3 2505.0000
LT.LDB10 3570.0000
LTS.CDB12 3030.0000
LI.TRA05 0.0000
LI.TRA06 0.0000
LA1.TRA07 5.0000
LT.TRA10 1.0000
LT.TRA20 0.0000
LT.TRA30 0.0000
LA.FTRAN -40.0000
LA.RTRAN -5.0000
LA.UMOD 20.6750
LA.TYRSER 1265.0000
LA.TYRPAR 2952.5000
TRPHOFF 2999.9998

Figure 3. A running synoptics application

D. Logs

On the old Control system several log programs are available. The logs read the parameter names of the equipment who's values are to be logged from a file . Then they acquire the current acquisition values for each of these parameters and write them to a file which is consecutively spooled to a printer and then deleted. In order to get Logs to the Workstation the Log program is remotely executed on the PDP system and the resulting file is transferred before being spooled. Finally the file may be displayed on the screen or sent to the local printer.

E. Application programs

Several more complex application programs could not be ported with the synoptics editor because of their graphical representation of the acquired data or because of their more complex data treatment. These programs had to be ported using directly the MOTIF development tools. For the moment this has been done for the Trace3d, a transfer line modelling program[2], the beam loss measurement along the Linac and the transfer lines and the spectrometry measurement. The static part of user interface has been specified in the User Interface Language (UIL) while the active part has been written in C (Motif initialization and callback routines). The emittance measurement programs are about to be ported. Here we tried to use an interactive user interface builder (Dec vuit) which generates UIL at its output. However a big improvement in development efficiency over handwritten UIL has not been seen, perhaps because of our inexperience using the tool.

V. CONCLUSIONS

The project was driven by two principal problems: The hardware consoles used at present in the Linac control room are becoming obsolete and difficult to maintain and access to the Linac control system was needed from the main control room. At present most of the daily operation work on the Linac can be done from the workstations in the MCR. Most of the programs developed so far will be reusable (with slight changes due to different equipment access especially for the repetitive acquisition) for the final system. We will have to finalize the beam diagnostics programs, generate many synoptics with the described tools and do the final switchover to the new system layout replacing the PDP-11 computers by DSCs.

References

[1] F. Perriollat, C. Serre, "Problems in rejuvenating a control-system and plans for the Cern Proton Synchrotron (CPS)", Nuclear Instruments and Methods in Physics Research A293 (1990) 81-86.

[2] U.Raich, "Beam Modelling with a Window Oriented User-Interface", Nuclear Instruments and Methods in Physics Research A293 (1990) 450-455.

[3] R. Rausch, Ch. Serre (Editors), Common Control System for the CERN PS and SPS accelerators, Nov. 1991, these Proceedings (ICALEPCS, Tsukuba, Japan, 1991).

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI10

A New Workstation Based Man/Machine Interface System for the JT-60 Upgrade

I.Yonekawa, M.Shimono, T.Totsuka and K.Yamagishi.
Japan Atomic Energy Research Institute
Naka-machi, Naka-gun, Ibaraki-ken 311-01, Japan

Abstract

Development of a new man/machine interface system was stimulated by the requirements of making the JT-60 operator interface more "friendly" on the basis of the past five-year operational experience. Eleven Sun/3 workstations and their supervisory mini-computer HIDIC V90/45 are connected through the standard network; Ethernet. The network is also connected to the existing "ZENKEI" mini-computer system through the shared memory on the HIDIC V90/45 mini-computer. Improved software, such as automatic setting of the discharge conditions, consistency check among the related parameters and easy operation for discharge result data display, offered the "user-friendly" environments. This new man/machine interface system leads to the efficient operation of the JT-60.

I. INTRODUCTION

The former JT-60 supervisory control system named "ZENKEI" consists of seven mini-computers and a CAMAC system for controlling tokamak machine operating conditions, discharge sequences and plasma equilibrium control.[1] Although been adequate the performance of "ZENKEI" has heavily worked, its limitations such as memory size, calculation speed, word length reached its limitation due to the option of the lower X-point operation and pellet injection system.

The new "ZENKEI" has to provide the long pulse operation (up to 15 sec), high speed plasma position feedback control (250µsec) and more user-friendly man/machine interface. [2],[3]

To satisfy the above requirements, we modified the plasma feedback control system of the two mini-computer system to that of a VME system. The man/machine interface system was changed from that of simple terminals to workstations. These workstations provide UNIX operating system with features of a multi-window system, network file system and many application tools for data handling and graphic interfaces. The improvements of the man/machine interface system have been made for the setting the discharge condition parameters, operation of the discharge and displaying of the discharge result data waveforms on the basis of the "friendly".

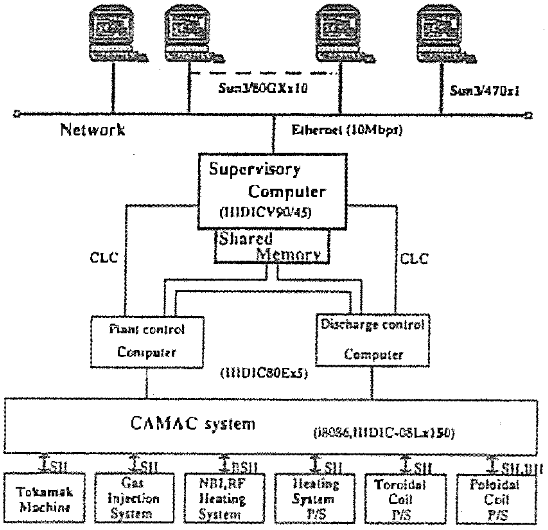


Fig.1 Man/machine System configuration in ZENKEI

II. SYSTEM CONFIGURATION

A. System Overview

The ZENKEI was composed of five mini-computers for discharge control and monitoring of plant operations. The CAMAC modules are used for data acquisition and control. Two mini-computers for high speed plasma position and shape control. The 14 CRT displays with keyboard and push-button switches had been prepared as the operator interface. This system had contained the following major functions;

- set discharge condition parameters,
- execute a discharge sequence,
- monitor the subsystem's operating status, and
- display the discharge result data.

The modification of many discharge condition parameters within a short period of time placed a big burden to the operator. Furthermore, the operator had to move back and forth from a certain console with a certain function to another with another. In addition, only a small memory had remained as a result of many modifications.

In order to improve the man/machine interface, computer hardware had to be replaced. The new man/machine interface system consists of 10 workstations as the operator's terminals. In addition one workstation acts as a file server. Every workstation replaces all 14 of the former CRT consoles. Figure 1 shows the new man/machine system configuration in ZENKEI.

B. Outline of the data flow

All data (including plant monitoring and discharge result data) are archived in a temporary file by the SVP (supervisor computer) from the HIDIC-80E (H-80E) system (former mini-computer system of ZENKEI) using data communication interface. There are two different communication routes between the SVP and the H-80E system. One is a DMA-controlled 16 bit, parallel interface. CLC (computer linkage controller) is assigned to small amount of data such as alarms and event signals. The other communication through a shared memory in SVP is assigned to a large amount of data such as discharge result (8Mbyte/shot).

Each workstation transfers a comparable large amount of data from a common file in SVP and SunSV (a server workstation) by using NFS (network file system) through the Ethernet.

TCP/IP protocol is also used for short data transfer, such as alarms and sequence event data.

The discharge condition is sent from a workstation to the SVP with NFS.

The discharge condition in SVP buffer file is sent to H-80E by the operator's mouse operation.

III. SYSTEM CHARACTERISTICS

A. Design principles

The major functions of the man/machine interface system are mentioned in the previous section.

The requirement of the system is to fully utilize the resource in the H-80E system which had "survived" for five-year operation in JT-60. For example, concerning the discharge sequence control, when the operator hits the sequence start button on the workstation, the message is sent to the H-80E and kicks the discharge sequence logic in the discharge management computer (1b) through the SVP → CLC.

As for setting the discharge condition, drastic improvement had been required to reduce mis-set of the parameters. As for the discharge result data display, simple and more user-friendly operation had been also required. Another principle is to be "more flexible", because user friendliness is inevitably completed after frequent modifications. The new system must interface with H-80E system which is linked to each subsystem controller composed of JT-60. It minimized the cost of modification, and reduced the risk in the software development.

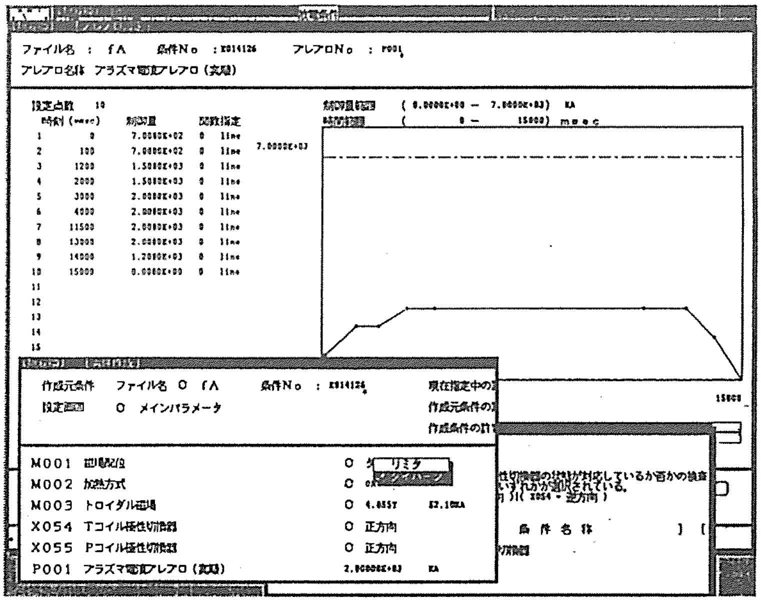
B. Setting of the discharge condition parameters

The requirements for setting the discharge condition parameters are a) minimize the number of parameters set by the operator shot by shot, b) set the parameters as easy as possible and c) hold consistency among the each discharge condition parameters. To satisfy the requirements we made the histogram of the parameters used among the previous 10000 shots.

As the result,the discharge condition parameters were found out to be classified into 3 groups corresponding to procedure to decide the parameters.

a) group 1; the parameters set by the operator directly.

The parameters such as plasma current, magnetic configuration (limiter/divertor), heating/joule experiment and intensity of the toroidal field were arranged as main parameters.



ファイル名 : fA 条件No : E014126 プレプロNo : P001
プレプロ名称 プラズマ電流プレプロ (実験)

設定点数 10

	時刻 (msec)	制御量	関数指定	
1	0	7.0000E+02	0	line
2	100	7.0000E+02	0	line
3	1200	1.5000E+03	0	line
4	2000	1.5000E+03	0	line
5	3000	2.0000E+03	0	line
6	4000	2.0000E+03	0	line
7	11500	2.0000E+03	0	line
8	13000	2.0000E+03	0	line
9	14000	1.2000E+03	0	line
10	15000	0.0000E+00	0	line
11				
12				
13				
14				
15				

制御量範囲 (0.0000E+00 — 7.0000E+03) kA
時間範囲 (0 — 15000) msec

作成元条件 ファイル名 O fA 条件No : E014126
設定画面 O メインパラメータ

M001 磁場配位 O リミタ
M002 加熱方式 O
M003 トロイダル磁場 O 4.855T 52.10kA
X054 Tコイル極性切換器 O 正方向
X055 Pコイル極性切換器 O 正方向
P001 プラズマ電流プレプロ (実験) 2.00000E+03 kA

Fig.2 Display of the preprogramed waveform, discharge condition parameters set by the operator directly and it guidance.

Hereafter,the parameters of the five poloidal field coils, such as the initial excitation current value and pre programmed current waveforms are arranged on the next screen. The gas injection valves and pre programmed pre filling, gas puffing waveforms are arranged in the same way.

b) group 2; automatic setting

The structural analyses and operational experience showed that quite a few discharge condition parameters can be uniquely decided by the group 1 parameters.Simple algorithms are developed in combination with the status or selection of devices set by the operator as group 1 items.

For example, when the operator selects the divertor configuration, the divertor coil power supply is automatically set by algorithm. The beam current parameters of NBI are automatically set by the beam acceleration voltage. The timing parameters of NBI control system are also automatically set by the data of the initial NB injection timing of the NB injection power waveform. The MG acceleration time is automatically set by information of the present rotating speed and the maximum currents of the coils. These algorithms are performed whenever the group 1 item is set.

c) group 3; fixed discharge condition.

The discharge condition parameters in the group 3 are changed only in modification of the system configuration and commissioning of the initial phase of the operation. It contains the data set of coefficient for the electro-magnetic probes, data sampling pitches and the value of operational limitation such as the maximum coil current and the vessel wall temperature.

Limitation/consistency check are performed when the operator selects the discharge condition parameters. And "user-friendly" consideration for displaying each discharge condition item such as guide for setting, menu selection , graphic user interface reduce the risk and load of setting the discharge condition. Using the mouse, the operator can access to a brief guidance for the discharge condition parameters; the names of items, rated value of setting, related consistency check and automatic setting algorithm.

After the completion, the operator may compare the complete set of the discharge condition with another to recognize the different parameters.

Anyone can make a different set of the discharge condition at the same time. A completed discharge condition file is registered to a buffer file.

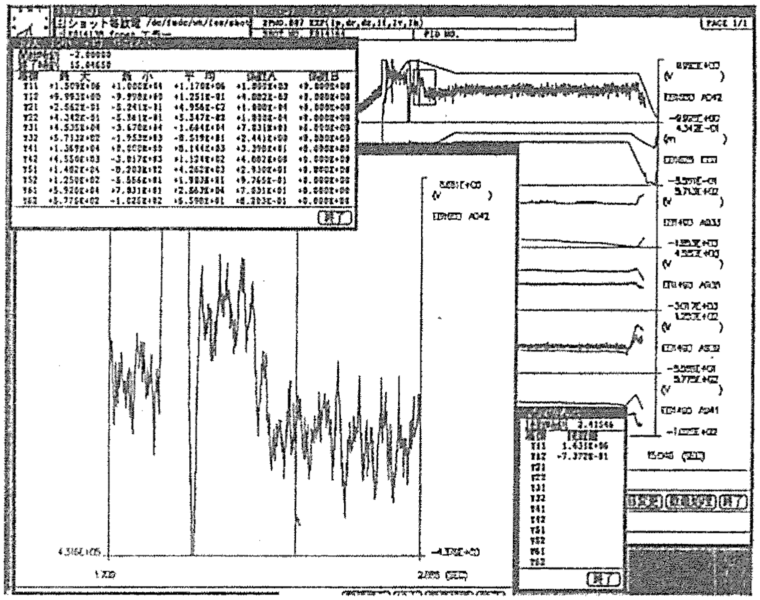


Fig.3 Discharge result data waveform with digital value display and enlargement of the waveform.

Then the chief physics operator selects a discharge condition and submit it to the next discharge sequence. The chief physics operator's approval is necessary to start the next sequence. Figure 2 shows the typical discharge condition parameter setting display.

C. Graphics

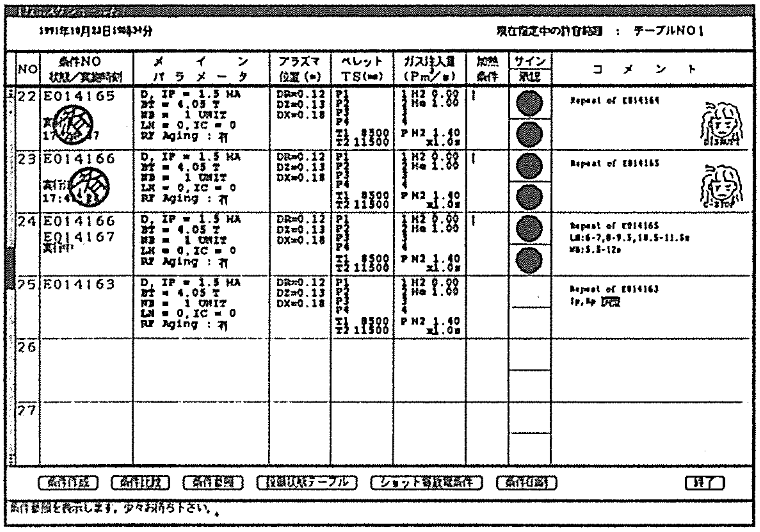
GKS and SunView software packages are used for graphics, because they have many tools for screen operation and multiwindow application. Figure 3 shows an example of the discharge result data display.

Icon selections are located in the lower position of a screen. They provide special functions; enlargement of the graphs, digital value display at the cursor(minimum, maximum , average values within the all sampled data or pointed out times by the mouse operation and calculated values of integration, differentiation through the pointed area), etc.

A graph specification table makes it possible to display the graphs quickly. The ID number of the measured data(PID.No) is listed in the utility file. The operator simply selects the PID.No to make the graph specification table.

Initially we had used simple text-based interfaces where the operator had to type using the keyboard.These primitive interfaces were replaced by using the SunView package. This enables the design of graphical interfaces with icons and menus.

The operator interface should be as simple as possible to prevent confusion. So, for instance, in mouse operation, the only left button is used for application software. Menu is also used for quick access to objective display.



1991年10月23日19時34分 現在指定中の許容範囲 ： テーブルNO1

NO	条件NO 状態/実施時刻	メイン パラメータ	プラズマ 位置(m)	ペレット TS(ms)	ガス注入量 (Pm³/s)	加熱 条件	サイン 承認	コメント
22	E014165 実行	D, IP = 1.5 MA BT = 4.05 T NB = 1 UNIT LH = 0, IC = 0 RF Aging : 有	DR=0.12 DZ=0.13 DX=0.18	P1 P2 P3 P4 T1 8500 T2 11500	1 H2 0.00 2 He 1.00 3 4 P H2 1.40 x1.0s	!	● ●	Repeat of E014164
23	E014166 実行 17:4	D, IP = 1.5 MA BT = 4.05 T NB = 1 UNIT LH = 0, IC = 0 RF Aging : 有	DR=0.12 DZ=0.13 DX=0.18	P1 P2 P3 P4 T1 8500 T2 11500	1 H2 0.00 2 He 1.00 3 4 P H2 1.40 x1.0s	!	● ●	Repeat of E014165
24	E014166 E014167 実行中	D, IP = 1.5 MA BT = 4.05 T NB = 1 UNIT LH = 0, IC = 0 RF Aging : 有	DR=0.12 DZ=0.13 DX=0.18	P1 P2 P3 P4 T1 8500 T2 11500	1 H2 0.00 2 He 1.00 3 4 P H2 1.40 x1.0s	!	● ●	Repeat of E014165 LH:6-7,8-9.5,10.5-11.5s NB:5.5-12s
25	E014163	D, IP = 1.5 MA BT = 4.05 T NB = 1 UNIT LH = 0, IC = 0 RF Aging : 有	DR=0.12 DZ=0.13 DX=0.18	P1 P2 P3 P4 T1 8500 T2 11500	1 H2 0.00 2 He 1.00 3 4 P H2 1.40 x1.0s			Repeat of E014163 Ip,Bp [illegible]
26								
27								

条件参照を表示します。少々お待ち下さい。

Fig.4 Shot schedule display

D. System design for graphical user interface

The effort of "quick response " in graphics and copies was made by applying the SPARC station in the network.

Another effort was made to design each "picture layout" in detail to satisfy the requirements of simpler operation.

For example, "shot schedule" screen provides entire information to all of the people in the control room , as shown in Fig 4. In this screen, the chief physics operator can modify the discharge condition parameters from the previous shot and able to confirm the main parameters and shot result. The duty manager decides to execute the discharge condition after the chief physics operator approves the discharge condition. After that the duty manager recognizes the discharge condition, together with both of the limitation check and consistency check of the error is detected, the sign becomes void.

In addition , the screen displays the progress of the discharge sequence. The status of the discharge sequence can be seen by changing the characters correspond to the progress of time sequence.

IV. LESSONS LEARNED

Some important lessons were learned from initial phase of. the operation.

- In a few tens of seconds of delay , the operator can not wait for the discharge display without irritation.

- Although the window size can be changed, it is seldom used. Icon is very useful and overlapping window is not so hard to see.

- Automatic photo copy of discharge result data display is seldom used. We provide automatic copy for quick information, but the operator just display objective graphs.

- The Japanese error messages system is useful for the operator. The detailed information is also available by the mouse operation.

-The remote maintenance function for each workstation has developed to maintain the system availability high enough.

V. FUTURE PLAN

More workstations are planned to be added to the network. It enables us to see the same information at the office as in the control room. Improvement of computer capability of the workstation and the SVP is desired.

Especially speed-up of the message transfer between the SVP and the H-80E is important for more quick response.

VI. SUMMARY

The new ZENKEI, workstation-based operator interface, has been operated for more than half a year. It is accepted by the operator using mouse operation instead of typing . An improved features of the new man/machine interface system are summarized;

1)The setting of the discharge condition parameters is drastically improved by adoption of automatic setting algorithm and fixed parameters.

2)The consistency check and graphical user interface reduce the risk and load of setting the discharge condition.

3) The shot schedule screen has been widely used as information; modification of the discharge parameters, status summary of the discharge sequence execution and discharge result summary.

ACKNOWLEDGEMENTS

The authors are grateful for many useful discussions with members of the control group; in particular , they would like to thank to Drs. M.Shimizu and T.Kimura for continuing support. The contributions of all other members are also gratefully acknowledged.

REFERENCES

[1] I.Kondo, et al., "The JT-60 Central Control System," Fusion Engineering and Design, vol.5, pp.69-84, 1987.
[2] K.Kurihara, et al., "JT-60 Plasma Control System, "Nuclear Engineering And Design, vol.11, pp.441-454, 1990.
[3] T.Kimura, et al., " VME Multiprocessor System For Control At The JT-60 Upgrade," IEEE Trans. on Nuclear Science,36 (1989) pp.1554-1558.

A Flexible Graphic Display System
for Accelerator Control

Cheol On PAK
Photon Factory, National Laboratory for High Energy Physics (KEK)
Oho, Tsukuba, Ibaraki 305, Japan

Abstract

A flexible graphic display system for controlling the KEK Photon Factory storage ring has been developed.

A VME computer locally controls the graphic display system and communicates with the host control computer through a RS-232C link. Graphic pictures are prepared in the local system by an interactive operation using either a tablet or a keyboard. The host control computer is free from any load due to graphics processing. In an on-line operation, pictures are displayed and modified by simple command strings from the host computer.

A "picture stack" method has been developed for this graphics system. The latest demanded picture always has top priority to be presented on each display monitor. Previous pictures are saved in a stack and can reappear when the current picture has been freed.

1. INTRODUCTION

Since colorful graphic displays can provide us with much useful information, even at a glance, they have become one of the indispensable tools needed for modern control systems. However, it often requires many man-hours to prepare a graphic display system, since graphics software is usually complicated and difficult to use. Furthermore, such graphics processing puts a heavy load upon control computers.

We have developed a flexible graphics display system (FGS) for controlling the Photon Factory storage ring. FGS has the following features:

- Host control computers are free from graphics processing load.
- Graphics are easily prepared without programming.
- Co-ordinate-free location method is possible.
- Any picture can be presented on any display monitor.
- Several pictures are kept in a stack manner for each monitor.
- Co-operative work with FTS (flexible touch screen system [1]) is possible.

2. SYSTEM OVERVIEW

A schematic of the FGS architecture is shown in fig. 1. The control system uses four minicomputers (FACOM S-3500 from Fujitsu) linked to each other by a token ring-type network [2]. The FGS control task (FGSCT) resides in one of the control computers. Many application tasks for control are distributed over four control computers. The application task sends a request to display a necessary graphic picture to FGSCT by DSM (Data Stream Manager, inter-task communication utility based on network [2]). FGSCT manages those requests from various application tasks. When it accepts a display request, it establishes a connection path between the application task and a proper display monitor. Hereafter, the application task is able to make modifications on a displayed picture. The connection path is valid until the application task frees this picture.

An intelligent graphic display station (DP-1000 from Digital) receives commands from FGSCT and draws pictures. One DP-1000 can handle three independent display monitors. The link between S-3500 and DP-1000 is RS-232C.

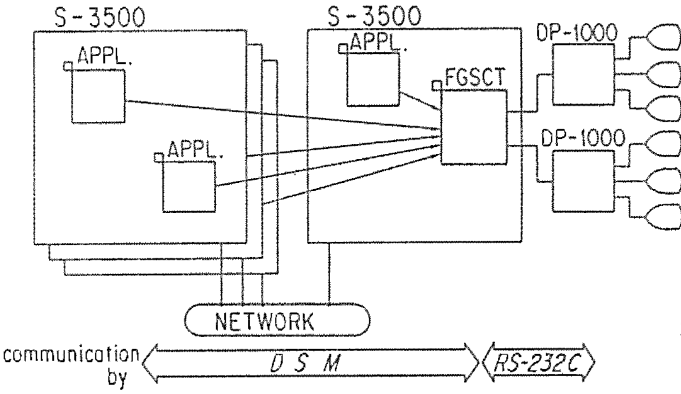


Fig. 1. Schematic of the FGS architecture.
APPL.: Application Task
FGSCT: FGS Control Task
DSM: Data Stream Manager (Inter-task communication based on token ring network)

3. FUNCTION OF FGS

A basic concept of FGS is that intelligent graphic display stations can present pictures by commands from control computers. Since the display stations perform all of the graphic processes, the control computers have only to send short command strings. Graphic pictures are prepared beforehand by the display station in a stand-alone manner.

Pictures are drawn interactively using a tablet, like a CAD; completed picture data are saved into static memories of the display station. A graphics screen consists of a basic picture and additional modifications of graphic element attributes such as color, blinking, or a new plot on a graph. Application tasks can choose any monitor to display. Usually, the FGS control task automatically assigns the nearest display monitor to the touch screen which made the application task start.

Another feature of FGS is a "picture stack" method. Fig. 2 schematically shows how the picture stack works. When an application task requests a new picture on one of the monitors, the old picture, if already there, is pushed into the picture stack. The last requested picture always has the highest priority to be displayed (it is called to be "active"). The pushed-in picture is not displayed, but can still be modified by commands from the application task. It is restored when the active picture is removed by a free command. Each display monitor has its own stack of eight layers. Graphic display stations have no responsibility with stack management, though the FGS control task in the control computer does. The FGS control task keeps all information concerning picture stacks and applied modifies.

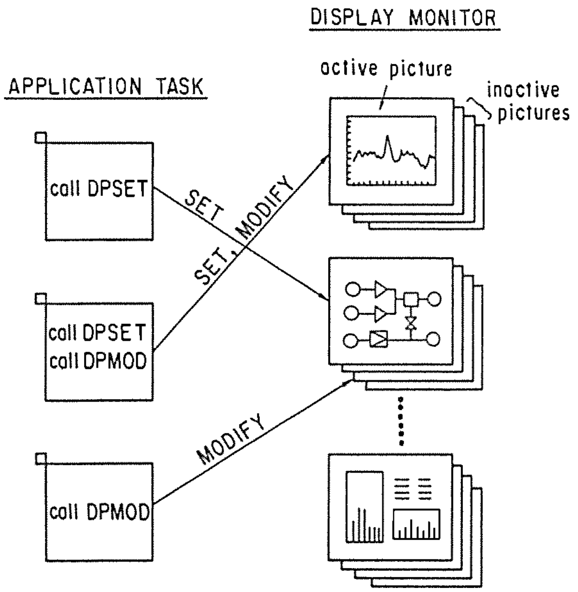


Fig. 2. Picture stack method of the FGS.

4. INTELLIGENT GRAPHIC DISPLAY STATION

DP-1000 is a VME computer system that was specially made for graphics processing by Digital Electronic Corporation. It has 4 Mbyte screen memories and drives three independent 20" CRT, each with a resolution of 1280 (horizontal) by 1024 (vertical) dots and 64 colors. It also has 5.5 Mbyte static memories for storing graphic picture data; this capacity corresponds to 550 standard pictures.

DP-1000 has two operation modes, local and remote. In the local mode, pictures can be drawn interactively using a tablet; no programming is required. Completed pictures are saved into memories. One of the unique features of DP-1000 is a "tag method", a co-ordinates-free location method. By tagging a graphic point(s) with a name, it is possible to point out pre-defined locations only by calling the tag names. Tag names can also include informations concerning graphic attribute, such as color, blinking and font type in addition to co-ordinates. In the remote mode, the host computer is able to select reserved picture data and to display it on a CRT by sending a simple command string to DP-1000 through an RS-232C link. Modifications on a displayed picture are also easily done by sending commands; the tag method is effective in this case.

5. SOFTWARE DESCRIPTION

5.1. *Software Interface to Application Program*

One feature of FGS is that an application program is simply and easily built. FGS serves several FORTRAN subroutine modules as follows:

- DPINZ initializes the FGS environment, called once at the start of the real-time system.
- DPSET selects a reserved picture and displays it on a monitor. The FGS control task sets pointers of the control tables, allocates the parameter area and establishes a bind between the application task and selected monitor.
- DPMOD modifies the picture. Modifications are effective for both active pictures (now on display) and inactive ones (pushed into a stack).
- DPFRE terminates a picture display and sets the monitor free to other inactive pictures, if any.

Only these four subroutines are sufficient for application programs to use FGS.

5.2. *Installation of New Pictures*

The way to install new pictures into FGS is as follows:

- Make basic picture on DP-1000 in the local mode using a tablet, and save completed picture data into memories.
- Develop modifications on the basic picture using an interactive test tool on a host computer, and store the final command strings into a modification record file on the host.
- Define a picture name and add the modification record file made in a previous step to the FGS command file.
- Install the FGS process into the application program using FGS service routines.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI11

6. CONCLUSION

FGS has given us an easy and speedy development of graphic presentation in the control system. It has also realized an effective use of limited resources of display monitors, owing to its picture stacking architecture.

We now have three display monitors with a single DP-1000, and are planning to add one more DP-1000; six monitors will eventually be available.

The FGS mechanism is strongly dependent on our hardware and software environment such as computers, display stations, network, operating system or programming language. Although it will therefor be difficult for other control systems to adopt FGS as it is, its conception may be useful for them.

7. REFERENCES

[1] Paper is in preparation.
[2] C. O. Pak, "The Control System of the Photpn Factory Storage Ring", Nucl. Instr. Meth. A277 (1989) 501.

Human-Machine Interface Software Package

D.K.Liu C.Z.Zhang
Institute of High Energy Physics
P.O.Box 918 Beijing

Abstract

The Man-Machine Interface software Package(MMISP) is designed to configure the console software of PLS 60Mev LINAC control system [1]. The control system of PLS 60Mev LINAC is a distributed control system which includes the main computer (Intel 310) four local station, and two sets of industrial level console computer. The MMISP provides the operator with the display page editor, various I/O configuration such as digital signals In/Out, analog signal In/Out, waveform TV graphic display, and interactive with operator through graphic picture display, voice explanation, and touch panel. This paper describes its function and application.

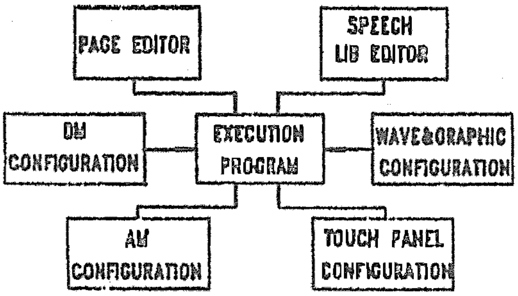


Fig.1 The Structure Diagram of MMISP

I. INTRODUCTION

Recently, the control level has grown up so fast by VLSI technology development. There are many kind of Workstion developed for interactive tool between operator and computer. Of course it has more powerful function but is also is expensive and can't connect a small system easily. We introduce a interactive software which is highly cost-effective, compact, and emphasizing easy operation based on the PC.

II. SYSTEM STRUCTURE

The MMISP shown as Fig.1 includes seven subroutines which are the Page Editor, the Speech library Editor, the Digital Monitor(DM) Configuration. the Analog Monitor(AM) Configuration and the Execution Program.

A.Page Editor

The page editor is used to edit the display picture and to create the drawing library. Its main function is follows:

* Drawing the line, circle, block line by cursor or up/down, left/right key

* 16 color could be selected

* The Picture can be moved, copied and loaded in hard disk as subpicture page

B.I/O Configuration

1. The DM configuration is used to create the display message of digital signals for the user's page, and the message will be saved in the page setting file. It has 5 kinds of digital display mode, which are the painting given area, the character string display, the drawing element display, the turn to the given page and the speaking something.

2. The AM configuration is applied to generate the display message of analog signals for the user's page, and the message will also be saved in the page setting file. It has six kinds of analog display modes, which are the digital display that the digit number can be selected from 1 to 7, the rectangle or other shape image display, the pointer meter display, and the turn to given page or the speaking something if the analog signal is overvalue.

3. The Wave & Graph Configuration is applied to create the display message of signal waveform, such as pulse voltage wave, or TV image for the user's page, and the message, such as the coordinate and display color etc., will be saved in the page setting file.

4. The Touch Panel Configuration is used to define the function of touch area, and to save these definitions in the page setting file. It has ten functions which are the recovering original color of given area, the changing color of given area, the input and display for a character string or data, the making a character string or data available or

refresh, the adjusting analog signal value by touch panel or encoder, the creating system control command, the entering a submenu, the exiting to a last menu, the turning to a pointed page and the calling user's program, DOS command or speech library.

C.Speech Program

This program is designed for explanation of various specification, machine operation such as operation guider. Also it can be used for warning some events such as "High Voltage Current is over, Please pay attention!".

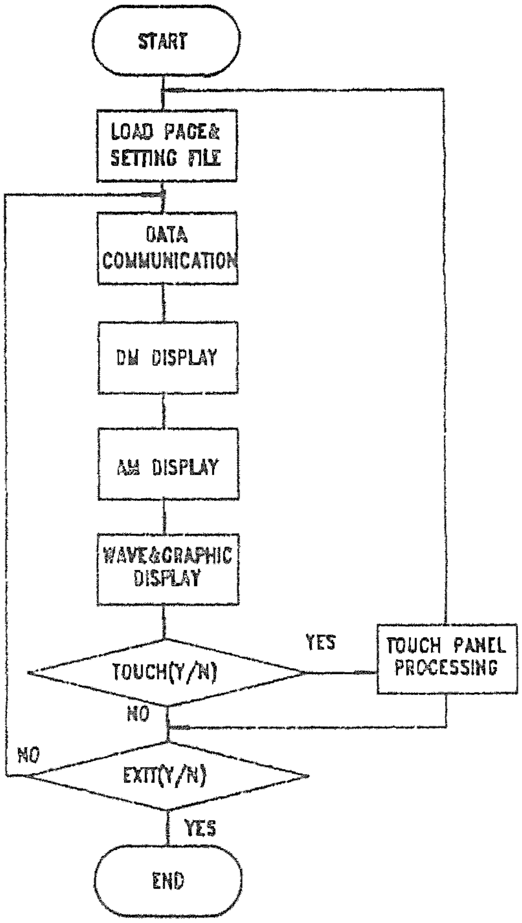


Fig.2 The Flow Chart of Execution Program

D.The Executive Program

This program is the core of MMISP. It is of six parts, shown as Fig. 2. These are the load page & setting file, the data communication with the central computer, the DM display, the AM display, the wave & graph display and the touch panel processing. The load page is to display the user's page and to get the configuration information from the page setting file. The data communication is to acquire the datum of digital signal and analog signal from the database of central computer.

The DM display is to process and display the digital signal according to the user's setting message from setting file. Similarly, the AM display is to process and display the analog signal or TV graph color code signal from the database of central computer, then to process and display at the region which the user defined in the setting file. The touch panel process is to respond to operator's instructions, to make the special treatment according to the user's definition in the setting file, for example, it can turn to the other page, or entry the sub-menu (Each page has 98 submenu, on screen can display 4 submenus at the same time), return to the last menu, and send the control command to central computer etc.

The Executive Program has three running modes, these are Test, Review and On-line Test. At the Test mode, the program generate every kind of data for checking the user's configuration. At the Review mode, the program provides the user to review the previous operation datum; and at On-line Test mode, the program obtains all kind of data from the database of central computer, and refreshs the data display every second.

III. APPLICATION

In general, an accelerator physicist can write the best program to resolve his physics problem as he understands better than software people. Therefore, he must study a long time about system configuration software and then he should understand the data path from controlled equipment to database and console display. But sometime, it is difficult for those specialists or physicists who are not familiar with system configuration software. They wouldn't like to resolve his problem by paying too much time to understand the whole configuration software. Considering this reason, comparing various interactive methods and our experience of accelerator operation, a set of utility interactive tools are used for control system of PLS 60Mev LINAC. The hardware environment of console interface is illustrated in the Fig.3. In the normal case operator can interact in four ways:

* Digital command could be through touch panel

* Analog control are adjusted by digitalized encoder or touch panel

* Various digital signal and parameter of accelerator are shown on the graphic screen.

* The voice explanation are used for warning some emergency events to operator.

The major procedure of system configuration using MMISP as following:

1.Picture edit

Graphic picture displaying various physical requirements can be edited by PAGE EDITOR easily just as in CAD. At first, designer should make definition of index page and chapters such as dividing modulator-klystron chapter, beam diagnostics chapter etc. Second, to indicate those region which show parameters and control buttons using PAGE EDITOR.

2.Setting I/O signals

Since every signal has its own system signal name, it is easy to point to a position on the designed page. It includes status indicate (DM), status control (DC), analog parameter display and control.(AM and AC)

3.Waveform and TV graphic setting

In the normal accelerator control, there are many waveforms such as klystron pulse voltage, pulse current etc. It very easy to configure and record as storage oscilloscope.

4. The function of voice explanation is useful to remind or warn operators to pay attention to accelerator.

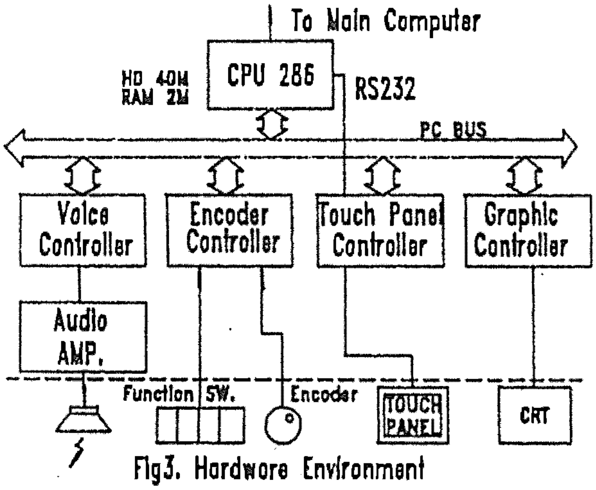


Fig3. Hardware Environment

IV. CONCLUSION

On other hand, this MMISP could be used in any control system as a movable console which can be linked to computer network using a communication board inserted to PC bus.

Summarized performance of MMISP is follows:

* Easy operation as CAD software

* Five kind of digital signal display; 150 signals could be used for one page

* Six kind of analog signal display; 150 signals could be used for one page

* Ten function of touch panel definition; 150 touch region for one page

* 10 waveform and 10 graphic display (30K points/1 picture)

* 98 sub-pages for one main page

* Easy to add new page into edited user system on real-time

V. REFERENCE

[1] "Instrumentation & Control System for PLS-IM-T 60Mev LINAC", Contributed to control conference November 1991

[2] K.Nakahara, I.Abe, K.Furukawa and N.Kamikubota, "An Operator- Console system of the Photon Factory Injector LINAC" Nucl. Instr. and Meth. A293 (1990) 446-449

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI13

Correlation Plot Facility in the SLC Control System*

L. Hendrickson, N. Phinney, L. Sanchez-Chopitea, and S. Clark
Stanford Linear Accelerator Center, Stanford University, Stanford, CA 94309, USA

Abstract

The Correlation Plot facility is a powerful interactive tool for data acquisition and analysis throughout the SLC. This generalized interface allows the user to perform a range of operations or machine physics experiments without the need for any specialized analysis software.

The user may step one or more independent parameters, such as magnet or feedback setpoints, while measuring or calculating up to 160 other parameters. Measured variables include all analog signals available to the control system, as well as calculated parameters such as beam size, luminosity, or emittance. Various fitting algorithms and display options are provided.

A software-callable interface has been provided so that a host of applications can call this package for analysis and display. Such applications regularly phase klystrons, measure emittance and dispersion, minimize beam size, and maintain beam collisions at the interaction point.

INTRODUCTION

Early in the development of the SLC, a generalized tool was written to acquire online data, and perform analysis and display functions across a wide range of information for many users. Rather than develop similar pieces of code for each combination of data, the Correlation Plot facility was designed generically to handle all of the data types available, and be extensible to other types that might evolve.

Due to the initial success of this implementation, a software-callable interface was added, so that other packages could make use of these fitting, plotting, and display facilities. This approach avoided redundant developments and provides a more consistent user interface for other parts of the control system.

ORGANIZATION

The main elements of the Correlation Plot facility are:

- ◇ A general control package which can step through setpoints of magnets, klystrons, feedback loops, timing parameters, and other device points of interest.
- ◇ A general data acquisition facility that can acquire data from a variety of sources, including high level parameters derived from analysis of klystron fast time plots and wire scans.
- ◇ A range of curve fitting algorithms, including average, linear, polynomial, sinusoidal, Gaussian, and specialized beam deflection curves.
- ◇ A general plotting package to display the acquired and fitted data. The sampled data may be plotted against the step variable, any of the sampled quantities, or the step number.
- ◇ A generic optimization feature allowing users to create a correlation plot to vary a step variable; obtain sampled data for each point, fit a parabola to the data, and implement the value of the step variable that results in the fitted minimum.

INTERFACES

Touch Panel

The Correlation Plot facility is an integral part of the SLC Control Program SCP [4]. The primary user interface utilizes a touch panel or cursor keys, although a mouse and trackball have been added as part of a newer X-window SCP. The main panel provides buttons for specifying the step and sample variables, selecting the range of the step variable, and setting other acquisition parameters. A generalized input parser interprets the input in a context-sensitive manner, where the meaning of each token depends on the valid tokens already accumulated. At any point, a list of the valid responses may be requested as a guide to the user.

From the touch panel or keyboard, the user may initiate data acquisition, terminate acquisition, or temporarily pause during an acquisition sequence. After data is acquired, display panels allow selection of fit and plot options. The user may request displayed or printed plots, as well as tabular formats. It is possible to specifically include or exclude selected data points, and have the facility recalculate the fit parameters.

An auxiliary output panel allows extended use of the system. Thus users may save data to disk files in various formats for offline analysis. Alternatively, users can reload previously stored command strings, or variables and data files, for further online analysis and display.

Callable Routines

All of the actions that are accessible via the operator interface are also available to software control. This makes it very easy to develop a layered application, using well-established building blocks. Callable functions support setting up variables and data acquisition options, and automatically acquiring desired data. Applications may obtain acquired data, perform a fit and retrieve the fit parameters, or provide for a variety of displays and plots. Some applications acquire data through specialized protocols, and then use the Correlation Plots for fitting and display functions.

---

*Work supported by Department of Energy contract DE-AC03-76SF00515.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI13

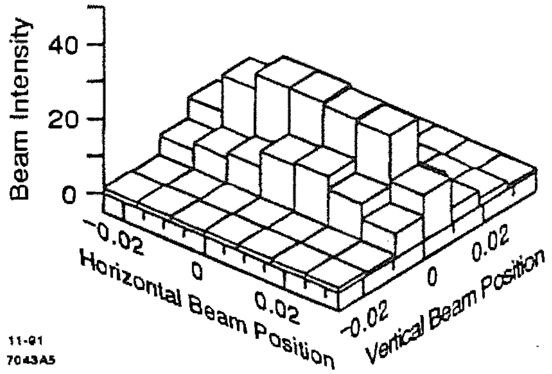


Figure 1. Beam aperture is studied by measuring beam intensity while varying horizontal and vertical position.

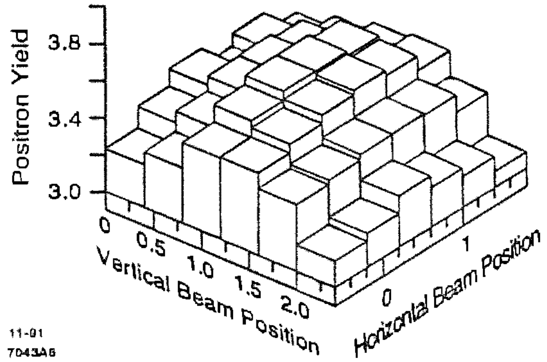


Figure 2. Optimization of beam position on the positron target.

CAPABILITIES

Monitoring

Correlation plot support is provided to measure or calculate a wide variety of data. Up to 160 variables may be sampled within a single acquisition sequence, and up to 100 data points are saved for each. At the current time, measured parameters include beam-related data from position monitors or toroids, and analog values from devices such as klystrons, magnets, thermocouples, and vacuum pumps, etc. For klystrons, in addition to simple analog values, the user may sample values derived from an analysis of the 64-pulse Fast Time Plot, such as phase and amplitude jitter, energy gain of the station, or perveance. This makes it possible to quickly scan the energy gain as a function of klystron phase to find the optimum setting, or to map out buncher jitter as a function of phase-shifter setting in the SLC injector.

Other calculated quantities available include energy, energy spread, particle yield, and beam position or deflection angle at the interaction point. Residual dispersion at the collision point may be measured noninvasively by correlating position and angle at the impact point with energy fluctuations. In addition, interfaces to other applications allow sampling of various derived quantities, such as beam states calculated by feedback and beam sizes, emittance and skew parameters determined from wire scans, beam scans, or profile monitor digitization. These quantities are used in a wide variety of beam optimization procedures.

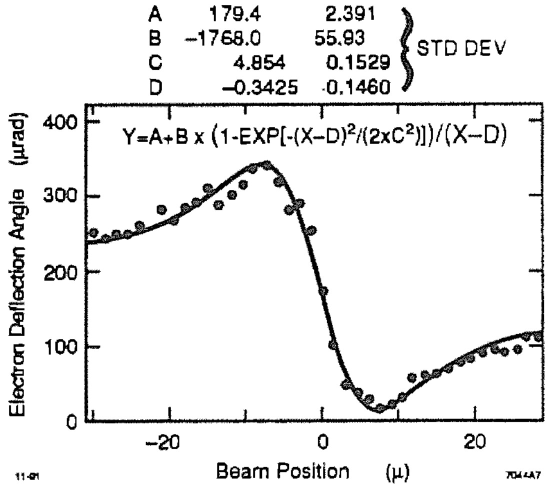


Figure 3. Beam-beam deflection fit at the interaction point.

Control

The user may elect to use either one or two step variables. Most of the variables available to software control have been implemented. Examples include:

- Setpoints of magnets or other analog control devices;
- Klystron setpoints, including amplitude, phase, and timing;
- Timing delays for any triggered device;
- Combinations of devices through the Multiknob facility;
- Setpoints of feedback loops stabilizing the beam [3,1];
- Time.

For many experiments, the Time step variable provides a simple delay between samples, in order to study the time structure of variations in normal running. Users can study correlations between sampled variables without modifying any control parameters. Most of the time, only one step variable is used, so a third has not been considered necessary. When two step variables are used, they define a grid of values, and the second is stepped through the whole range for each setting of the first. Figures 1 and 2 are examples of plots with two step variables.

Data Reduction

To aid in the analysis of the data, a variety of fitting routines may be selected. The selection of fitting algorithms may be accomplished through the user interface (touch panel) or by application software. Figure 3 shows the special beam-beam deflection fit used for optimizing collisions and estimating beam cross section at the interaction point.

APPLICATIONS

Various software applications used in the SLC have been built upon capabilities of the Correlation Plot facility. These analysis and control applications include the following:

1. Multivariate correlation plots make it possible to visualize a wealth of data in a focused and informative way. Figure 1 is a plot from a beam aperture study in which multistep variables are used to study beam intensity in three-dimensional space. Figure 2 is an optimization study of beam position on the positron target, where

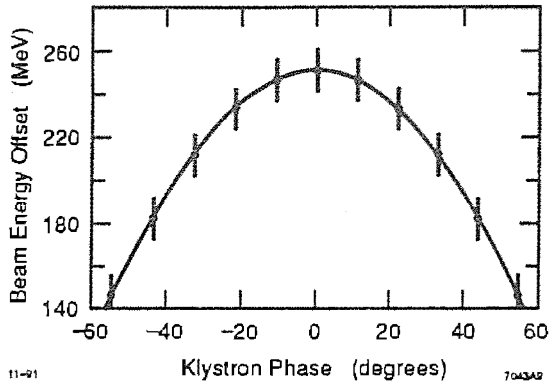


Figure 4. An automated procedure optimizes phases of the 240 SLC klystrons.

feedback setpoints are used to control horizontal and vertical beam position, and the application calculates positron yield at each combination of setpoints.

2. Data collection and optimization applications that calculate optimized values for klystron or magnet settings are common. Figure 4 displays the fitted beam energy offset for a single klystron, where an automated procedure steps through the devices, measuring beam energy as a function of phase in order to determine the optimum setting. Figure 5 displays a parabolic fit of beam width squared, from digitized profile monitor data as a function of quadrupole magnet strength.

CONCLUSIONS

The Correlation Plot facility has proven to be an extremely powerful tool for the analysis of accelerator functionality, device commissioning, and for building software applications. The flexibility provided by the different types of variables that may be controlled and monitored has allowed operators and physicists to rapidly design and execute a vast assortment of experiments without any new or specialized software. In fact, the Correlation Plots are used so extensively that most experimental data presented comes from this facility, and it is extremely rare that data needs to be plotted offline. Even for complicated experiments where further analysis is required, the Correlation Plots provide the data acquisition and online validation. The Correlation Plots facility has made an essential and invaluable contribution to the SLC development.

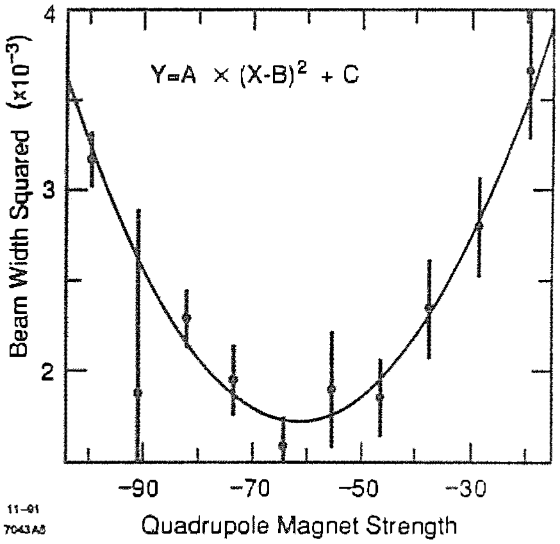


Figure 5. Beam emittance is calculated by varying a quadrupole setting and fitting a parabola to the estimated beam width squared.

ACKNOWLEDGMENTS

The authors wish to thank Julian Kupiec, who wrote the original package, and Miguel Flores, who added many features.

REFERENCES

[1] K. Thompson et al., "Feedback Systems in the SLC," Proc. 12th Particle Accelerator Conf., Washington DC, March 1987; SLAC-PUB-4217.

[2] M. C. Ross et al., "Wire scanners for beam size and emittance measurement at the SLC," Proc. 1991 IEEE Particle Accelerator Conf., San Francisco, CA; SLAC-PUB-5556.

[3] T. Himel et al., "General, Database Driven Fast Feedback System for the Stanford Linear Collider," Proc. 1991 IEEE Particle Accelerator Conf., San Francisco, CA; SLAC-PUB-5469.

[4] R. E. Melen, "Centralized Digital Control of Accelerators," Proc. Europhysics Conf., Berlin, West Germany, September 1983; SLAC-PUB-3218.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S13MMI14

ICONIC REPRESENTATION OF PARTICLE BEAMS USING PERSONAL COMPUTERS

S. Dasgupta, D. Sarkar, C. Mallik
Variable Energy Cyclotron Centre, 1/AF, Bidhan Nagar, Calcutta-64

Abstract

The idea of representing the character of a charged particle beam by means of its emmittance ellipses, is essentially a mathematical one. For quick understanding of the beam character in a more user-friendly way, unit beam cells with particles having a uniform nature, have been pictured by suitably shaped 3-D solids. The X and Y direction momenta at particular cell areas of the particle beam combine together to give a proportionate orientation to the solid in the pseudo 3-D world of the graphic screen, creating a physical picture of the particle beam. This is expected to facilitate the comprehension of total characteristics of a beam in cases of online control of transport lines and their designs, when interfaced with various ray-tracing programs. The implementation is done in an IBM-PC environment.

INTRODUCTION

The practice of representing particle beam in terms of the phase-space figures at an axial location of the beam, is well-established. The phase-space diagrams can be either a plot between x/y and p_x/p_y, where p_x and p_y denote momenta in x and y directions respectively. Consequently these 2-dimensional figures, are used to convey informations about an entity which exists in reality, in a 4-dimensional phase-space consisting of x, p_x, y and p_y as the dimensions. Beam-line designers as well as operators optimizing transport of particles through beam-line elements, very often refer to these 2 dimensional projections idealised to ellipses, to study and optimise the transport of a particle-beam. Any of these ellipses, eg. x vs θ (= p_x/p, where p is the longitudinal momentum), though conveys the information about distribution of a particle-population with specific ranges of x momenta, as a function of x, but it does not immediately produce any idea about their correlations with the y-axis. Conversely, the same inadequacy applies for the y vs ϕ ellipse. Therefore as far as qualitative understanding of the beam is concerned, as a first impression, views of the ellipses are not complete enough. The mental process of a person doing the optimizations has to be only analytical, which is not a very comfortable situation. Consequently, it was thought that a more expressive diagram, which will convey a qualitative idea about all the 4 dimensions of the beam-ellipsoid, should contribute as a more friendly feedback to the user of a transport-optimization procedure.

With the vastly expanding use of computers with image graphics capabilities by transport-line designers and graphic workstations as operators' consoles in accelerator control, the 3-D graphic generation capabilities of these computers, can be tapped in such a situation. The particle beam, if could be made visible together with all its angle,

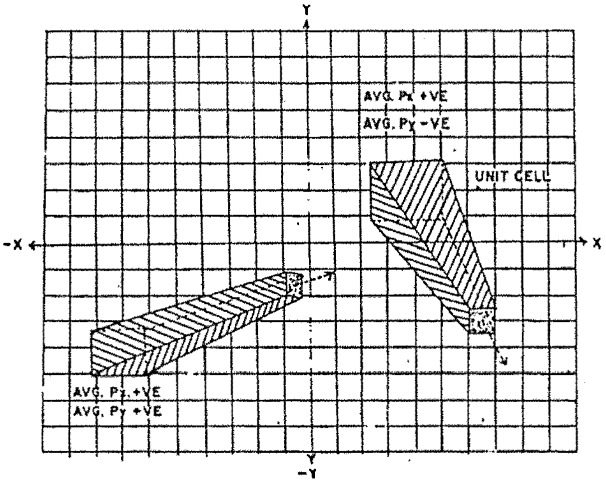


Figure 1. Two beamlets with different momenta

should give a feel and a consequent instantaneous understanding of the beam-parameters to any kind of user.

IMPLEMENTATION-PLAN

The total transverse cross-section of a beam can be thought of as composed of a large number of cells, through each of which a beamlet can be thought to be emitted. Particles occupying a particular cell are characterised by θ_{av} (ie. $p_{x(av)} / p$) and ϕ_{av}(ie. $p_{y(av)} / p$) and hence the direction of beamlet is represented by these two averaged quantities.

Now each beamlet can be represented by the figure of a solid (Figure 1) of the shape of a truncated rectangular pyramid. This bucket-like figure is capable of creating the impression of a directional movement of the beamlet. The presentation being 3-dimensional, the orientation of the beamlet in both x and y directions can be depicted. Reducing the arbitrary cross-sectional area of the cell and thereby increasing the number of beamlets over the transverse section of the beam can create a picture in more detail.

The number of cells can be optimally chosen to produce a suitably accurate picture with sufficient details and yet at the same time care is taken so that the buckets do not obscure one another to a large extent. In order to bring home the effect of orientation of the buckets in 3-dimensional space, the four lateral surfaces and the front surface of the buckets are all to be painted in five distinctly seperate colours (Figures 2 & 3).

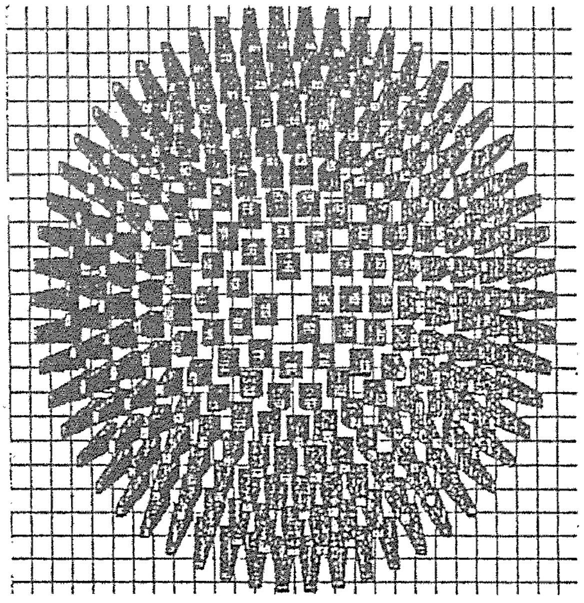


Figure 2. A diverging beam

PROGRAM-DETAILS

The data about the particle density and their momentum distribution, as obtained either from the on-line emittance measurement devices or from transport optimization codes (eg. TRANSPORT, TURTLE, GIOS etc.), is assumed to be present in disk file. The data file should be actually re-organized as an array of records, each record having four fields.

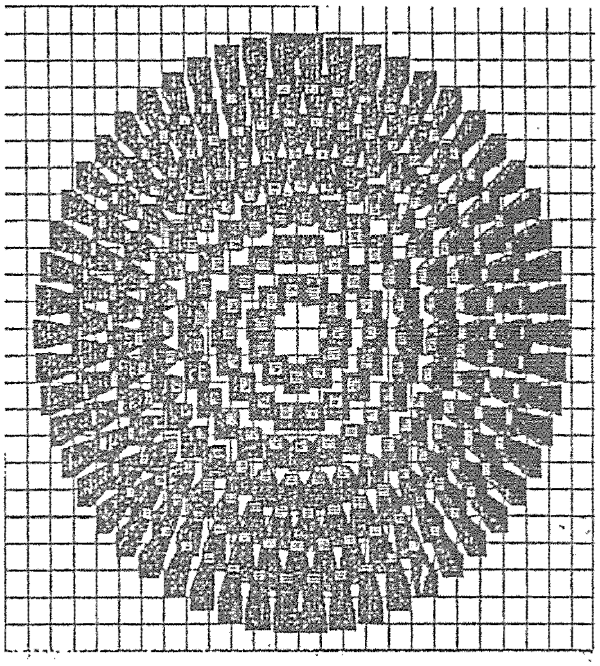


Figure 3. A converging beam

The field designators are as follows.

posn : { Array of X and Y co-ordinates of the centre of a unit cell on the transverse plane of the beam }

xmom : { Computed average momentum in x-direction of all the particles in the unit cell }

ymom : { Computed average momentum in y-direction of all the particles in the unit cell }

density : { Total number of particles in the unit cell}

Some additions in the standard transport calculation programs or emittance-data acquisition programs are done to organize the data in above format.

In reality, the momentum angle variations are typically in the range of 100 milliradian, a much too small quantity to be visually resolvable. Hence, for the purpose of our picture-generation, this quantity has to be scaled up by a suitable factor (typically 300) and a proportionate variation in the angles of buckets can then be made recognisable on the screen.

In the present version, in absence of 3-D graphic generation routines, procedures were written in Turbo-Pascal language to independently generate the X and Y-rotation effects on the bucket and then to combine to create resultant solid figure. Standard algorithm was utilized to eliminate drawing of hidden surfaces of a bucket at various orientations [1].

It was felt that though perspective views of the buckets can give a clearer perception, but with a little experience this loss of visual perception will not be too damaging, to warranty taking up these time-consuming computations in a personal computer. The other problem was to keep the consistency of depth-perception when buckets of varying orientations, overlap one another. In other words, the sequence in which buckets should be drawn would have to be a difficult and lengthy algorithm, in absence of any depth information of the image. This information is not kept in the present version, since insufficiency of frame buffer memory in our PC does not allow any benefit from this information, during 3-D image generation. A simpler scheme has been devised to give satisfactory results. Based on the relative co-ordinate displacement of the tip of the bucket with respect to its base, the bucket parameter records are divided into two groups, named arbitrarily as 'converging' and 'diverging'. Then each group is internally sorted according to increasing radial distance of the centre point of the base of the bucket. For 'converging' group, buckets are drawn starting from the beam-centre of the cross-sectional area towards the periphery whereas for the 'diverging' set, they are drawn in the reverse sequence. As far as the individual sets are concerned, buckets within that set overlap in the right way.

FUTURE EXPANSION

A 'zoom' option will allow a progressive blowing up of the figure, enabling more detailed observation of selected zones of the beam-section. This option will be helpful if the buckets fill up the space too densely so as to obstruct view of the buckets in the background.

A cross-hair cursor (figure 4) can be taken to any point on the figure and clicked. There would be two output windows which will show the particle population distribution along the x-axis and that along the y-axis of the current cursor position, thus creating the scope for realisation of relative distribution of particles also. Two other output windows

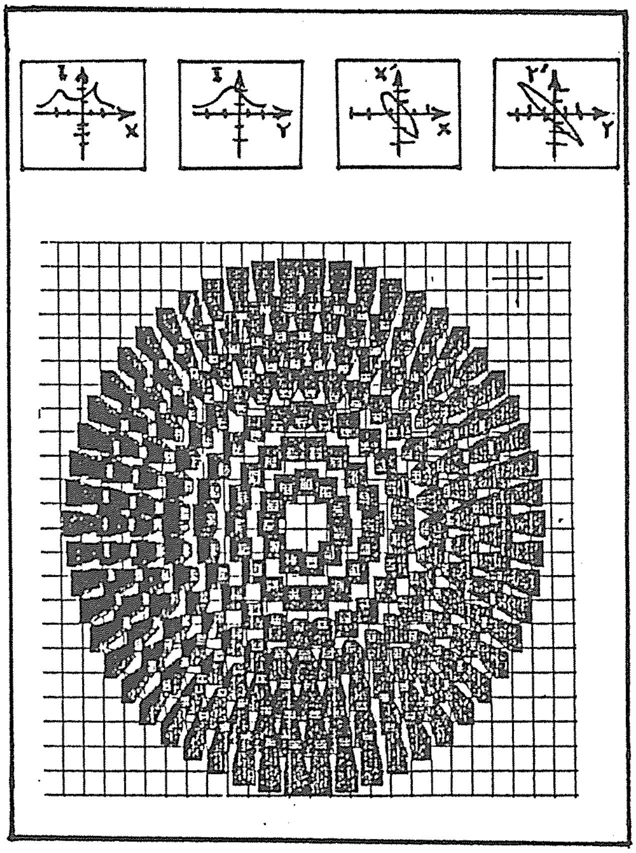


Figure 4. An integrated beam-view

can give the output in the form of the standard phase-space ellipses, for the total beam at the particular longitudinal location in the beam-line, thus creating an integrated view of all parameters of interest of a particle beam.

In suitable cases, animated movie-frames [2] can be generated to simulate the dynamically changing beam characteristics, because of disturbances introduced naturally or otherwise.

DISCUSSIONS

The idea of the development is to incorporate the comparatively easier accessibility of 3-D graphics in the user-interface area of certain computer programs used in accelerator design and control, and to re-inforce the standardized outputting style with a more physically revealing style. The results obtained in the current work, though, are sufficiently useful for trained eyes, yet for those who can afford using powerful graphic workstations, they can apply the same idea and can get more realistic 3-dimensional views.

Such views are expected to give an immediate impression about a beam at any place and can trigger the intutive process of manipulative operations necessary on the beam. In other words, the idea is a step in the line that a computer should provide the designer as well as the operator, facilities to visualise the beam in an integrated way to closely match his own way of percieving the beam.

ACKNOWLEDGEMENTS

The authors are thankful to Dr. P.K. Sarkar for some active discussions and fruitful suggestions. Shri R.B. Bhole has very laboriously helped in testing and debugging of the codes. We shall like to thank him.

REFERENCES

[1] N.M. Moris, *Computer Graphics & CAD Fundamentals*, BBC Microversion, p.159

[2] Alfredo Luccio, "Algorithms and Modelling For The AGS Booster", Nucl. Instr. and Meth., Vol. A293 (1990), pp.460 - 463

OBJECT-ORIENTED PROGRAMMING TECHNIQUES FOR THE AGS BOOSTER*

Joseph F. Skelly
AGS Department, Brookhaven National Laboratory
Upton, New York 11973

Abstract

The applications software developed for the control system of the AGS Booster Project was written in the object-oriented language, C++. At the start of the Booster Project, the programming staff of the AGS Controls Section comprised some dozen programmer/analysts, all highly fluent in C but novices in C++. During the course of this project, nearly the entire staff converted to using C++ for a large fraction of their assignments. Over 100 C++ software modules are now available both for Booster and general AGS use, of which a large fraction are broadly applicable tools. The transition from C to C++ from a managerial perspective is discussed and an overview is provided of the ways in which object classes have been applied in Booster software development.

Introduction

At the outset of the Booster Project,[1] management decided to promote the use of object-oriented techniques among the programming staff. Our hope was to achieve improved programming efficiency and greater maintainability of code through increased modularity. The C++ language was chosen because of its accessibility to a staff fluent in C, and because it was well supported on the computing system already in place. Whereas prior efforts at in-house self-education in C++ had yielded only very limited success, our staff now is very comfortable using C++, and we consider that our goals in promoting C++ have been satisfactorily achieved. During the past two years, our programmers have accumulated nearly 200 staff-months of experience with C++, and produced some 160 source-code modules totaling more than 100,000 lines; of these, more than 80 are tool modules which define more than 300 object classes. The Booster was commissioned in June of this year; during this period our software was exercised vigorously, and software performance and user reaction were favorable. The reasons for this success will be discussed below.

Environment

The AGS Distributed Control system (AGSDCS) comprises a network of approximately 50 Hewlett-Packard/Apollo workstation nodes on a Domain token-ring network which spans the AGS accelerator complex. Ten workstations provide the operator interface at five consoles in the AGS Main Control Room. About 15 workstations are used for programmer or physicist development nodes, and the remainder are used as control system consoles by engineering and technician work groups among the accelerator staff, or as data-collection servers in the accelerator complex. The workstations run a Unix-like operating system and provide a high-resolution display, for which an internal Graphics User Interface (GUI) standard for the programs has been established.

The AGSDCS is interfaced to some 5800 accelerator devices via more than 100 so-called "device-controllers" in more than 50 locations. The device-controller layer is currently implemented with Intel Single-Board Computers (SBCs) in Multibus packaging. Device-controllers are connected to so-called "stations" via the GPIB (IEEE-488) bus; stations are implemented either in Multibus SBCs (the older AGS version) or in Apollo workstations (the new Booster version). Access by high-level programs to the network of accelerator devices is supported by a library of toolkit routines which permit a device to be referenced by just its name. The library routines resolve the device address in the network by reference to descriptor tables constructed from a relational database which describes the entire control system.

Transition to C++

A number of factors are discussed here which contributed to the successful transition of the staff to C++. Experience with this process suggests that each factor is important, and that the absence of any one of them would have had a very negative impact on its success.

Assignment Profile

Staff members were given independent software assignments for the Booster Project, and permitted to develop them individually. The opportunity to nurture a new project from its inception without undue burden of prior development encouraged the staff to apply new techniques. In addition, it was recognized early that many of the assignments required common tools, and management fostered cooperative efforts

---

*Work performed under the auspices of the U.S. Department of Energy.

among the staff to define and develop generic object-oriented tool packages.

Staff Experience

The programmers had already mastered the details of the control system infrastructure. New staff hired for the Booster Project were given adequate time to become familiar with the control system before learning C++ and addressing the new Booster-specific programming assignments.

C++ Lead Programmer

The staff was seeded with one experienced C++ programmer to serve as in-house consultant and mentor. During the succeeding year, the staff members' primary responsibility became their Booster assignment, and as they addressed this assignment they adopted C++ as their design language. A C++ culture was established within one year, and a class library rapidly accumulated which functioned as a peer-developed resource of programming models.

Classification of Classes

The class library was recently examined to acquire a snapshot of its contents (which are still expanding). The contents have been categorized according to the type of services which the classes offer.

Class Category	# Modules	# Classes
Operating System Services	14	36
GUI Services	4	22
Control System Services	22	60
Data Acquisition, Display	17	53
Device Tools	12	28
Accelerator Tools	16	87
Accelerator Physics Tools	7	44

Although the class library contains a large number of classes, many of these are intended only for internal use by the tool modules; a programmer wishing to use these tools need become familiar with only a few classes at a time.

Class Examples

Some samples are offered of the classes in each category of the class library, along with some methods (function members) defined for them, in order to exhibit the ways in which these classes are applied. The format in this table is similar to the C++ code from which these examples were derived: a class-definition line ("class ClassName") is followed by a number of lines defining methods for the class ("ClassName::method"). The formal parameters (arguments) for the methods are not displayed, for the sake of simplicity; likewise, most of the syntax of the C++ language is suppressed, although class derivation is exhibited.

Accelerator Tools - A Special Niche

An object-oriented approach to design of the "accelerator tools" category seems to offer a special opportunity for programmers in an accelerator controls environment. It is often the case that the architecture of the control system imposes constraints on the hardware designers, constraints which cause the elements of the accelerator to be artificially fragmented into multiple "devices", or "control system primitives". In the AGS control system, the control system primitive is called a "logical device". As an example, the engineer designing an interface for a multi-wire profile monitor or "harp", was obliged to implement the timing control as one logical device, the gain control as a second logical device, positin control (insert/retract) as a third logical device, and acquisition of the profile as a fourth logical device. Moreover, gain and timing control were shared among a collection of several harps in the same beam line. This complexity is by no means unusual, and is a common consequence of the necessity to standardize control system architecture and to solve difficult accelerator design problems.

With an object-oriented tool to support program interaction with a harp, the complexities resulting from the multi-device interface can be hidden inside the class design. The high-level programmer can then interact with a single entity--the harp object--and function much the same way the physicist does when he views the harp as a single component of the

Table I. Class Samples - Operating System Services

```
Class  SharedMemory
    SharedMemory::GetLock
    SharedMemory::ReleaseLock
Class  MbxMessage                       //    Mailbox Message
Class  ApolloMail  :  MbxMessage        //    derived class from MbxMessage
    ApolloMail::ServerGet               //    server access to mailbox
    ApolloMail::ServerPut
    ApolloMail::ClientGet               //    client access to mailbox
    ApolloMail::ClientPut
```

Table II. Class Samples - GUI Services

Class PopupMessage : GenericPopup	//	derived class from Generic Popup
PopupMessage::display	//	display in proportional font
PopupMessage::display_mono	//	display in monospaced font
PopupMessage::display_ok	//	display and await confirmation
PopupMessage::ask_yn	//	ask question, get yes/no reply
Class PopuMenu : GenericPopup	//	derived class from Generic Popup
PopupMenu::getchoice	//	get a single choice
PopupMenu::getchoices	//	get multiple choices

Table III. Class Samples - Control System Services

Class Alarm		
Alarm::Log	//	Log in database
Alarm::DeLog	//	DeLog from database
Alarm::Priority		
Class Sld : Alarm	//	alarm for Sld (Simple Logical Device)
Class Controller : Alarm	//	alarm for Controller

Table IV. Class Samples - Data Acquisition, Display

Class SldRequest : DataRequest	//	derived class from DataRequest
This class is not exported to the public; it is used by DataCollector		
Class DataCollector		
DataCollector::settimeout	//	set timeout period
DataCollector::setup	//	set up list
DataCollector::get	//	request data, wait until it arrives
Datacollector::getimmediate		
Datacollector::getsynchronized		
Class GraphMonitor : Monitor		
GraphMonitor:resize	//	resize the graph
GraphMonitor::title	//	display routines
GraphMonitor::writelabel		
GraphMonitor::writecycle		
GraphMonitor::hardcopy	//	hardcopy to printer

Table V. Class Samples - Device Tools

Class FunctionGenerator		
FunctionGenerator::menu_edit	//	edit function
FunctionGenerator::load	//	load it to devices
FunctionGenerator::readback	//	read the devices
FunctionGenerator::set_cld_names	//	names of complex-logical-devices
FunctionGenerator::set_default_value		
FunctionGenerator::set_start		
FunctionGenerator::set_end		
FunctionGenerator::set_timing_cld_names		
FunctionGenerator::set_tolerance		

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S1400P01

Table VI. Class Samples - Accelerator Tools

```
Class  Instrument
    Instrument::calibrate
    Instrument::acquire_data
    Instrument::display_data
    Instrument::save_data
    Instrument::read_data
Class  HARP  :  Instrument                    //      multi-wire profile monitor
    HARP::insert
    HARP::retract
Class BPM  :  Instrument                      //      Booster Position Monitor
Class XF  :  Instrument                       //      Transformer
Class  MagnetCalibration
    MagnetCalibration::ReadCalibrationDataFile
    MagnetCalibration::Interpolate
    MagnetCalibration::ReadIvalues
    MagnetCalibration::ReadBvalues
Class  Transient Recorder
    TransientRecorder::GetLiveReadback
    TransientRecorder::SaveLiveReadback
    TransientRecorder::GetSavedReadback
    TransientRecorder::DisplayReadback
```

Table VII. Class Samples - Accelerator Physics Tools

```
Class  ManualHarmonicsCorrector  :  OrbitCorrector
    ManualHarmonicsCorrector::set_harmonic
    ManualHarmonicsCorrector::set_pue_display
    ManualHarmonicsCorrector::display_setpoint_harmonics
    ManualHarmonicsCorrector::display_readback_harmonics
    ManualHarmonicsCorrector::increment_coefficient
    ManualHarmonicsCorrector::execute_correction
Class  BoosterOrbitBump
    BoosterOrbitBump::magnet_device_list
    BoosterOrbitBump::pue_device_list
    BoosterOrbitBump::what_bump_order
    BoosterOrbitBump::what_bump_type
    BoosterOrbitBump::magnet_readbacks
    BoosterOrbitBump::magnet_measurements
Class  TuneModel
    TuneModel::WriteTuneIntoSetpoints         //      Send setpoints to devices
    TuneModel::ReadSetpointsIntoTune          //      Read setpoints from devices
    TuneModel::StartMad                       //      Run modeling program MAD
    TuneModel::TestMadDone
    TuneModel::GetTwissAtElement              //      Get Twiss params from model
    TuneModel::DisplayTwissAtElement          //      Popup Twiss param display
    TuneModel::DrawBeamLine                   //      Iconic display of beam line
    TuneModel::DrawEnvelope                   //      Draw beam envelope
    TuneModel::DrawAperture                   //      Overlay magnet apertures
    TuneModel::DrawPhaseEllipseAtElement
```

accelerator. This opportunity for class design to offer a clean interface to accelerator components is characteristic of these accelerator tools. With proper class design, a high-level program can be coded to read as cleanly as the designer's statement of the program function.

Acknowledgments

The work discussed here was developed over the last two years by the entire staff of the Controls Section of the AGS; their contributions made this report possible, and their cooperation made the work a pleasure.

Reference

1. W.T. Weng, Construction and Early Commissioning Results of the AGS Booster, 1991 Particle Accelerator Conference (in press).

A Simplified Approach to Control System Specification and Design Using Domain Modelling and Mapping.

G.A. Ludgate
TRIUMF, 4004 Wesbrook Mall, Vancouver, B.C., V6T 2A3, Canada

Abstract

Recent developments in the field of accelerator-domain and computer-domain modelling have led to a better understanding of the "art" of control system specification and design. It now appears possible to "compile" a control system specification to produce the architectural design. The information required by the "compiler" is discussed and one hardware optimization algorithm presented. The desired characteristics of the hardware and software components of a distributed control system architecture are discussed and the shortcomings of some commercial products.

I. INTRODUCTION

In recent years more emphasis has been placed on the gathering and validating of requirements for automated control systems before they are built [1, 2, 3, 4, 5]. Our earlier work reported on the specification of the KAON Factory Central Control System (KF-CCS), using two emerging techniques in the application of object-oriented principles to requirements specification namely: domain driven modelling [2] and dynamic object modelling [3]. A re-examination of the problems encountered using these two contemporary techniques has led to a better understanding of both the use of domains in creating and structuring a system specification and of the design-processes used to transform the system specification into executable code.

It now seems possible to "compile" a control system from its specification form. As with all "compilers" the "target language" (usually a micro-processor machine code) must be *exactly* described before the "compiler" can be created. With contemporary domain modelling approaches we now have the power to construct such a complete description of the active elements that form control systems and to determine an appropriate strategy for "compilation".

II. PROBLEMS WITH EARLIER APPROACHES

Dynamic object modelling [3] advocates, from the outset, the determination of the context of a system-to-be-built and its presentation in the form of a Context Diagram (Fig. 1). A Context Diagram follows from an analysis of a problem, the specification of a solution and the desire to implement the solution as an automated system. This approach leads to an early identification of external devices (Terminator Objects) that are to be interfaced to and controlled by the system. Only information flows between the system and the Terminator Objects are shown on a Context Diagram.The single bubble represents the system-to-be-built; boxes represent Terminator Objects.

The internal structure of the system-to-be-built, termed the Object Communication Diagram, is comprised of both the static and dynamic system-objects in the solution. Figure 2 shows an example internal structure for Fig. 1. Static objects are representation of conceptual entities in the solution (e.g. schedules, lists etc.) while each dynamic object represents the dynamic behavior of its associated Terminator Object, as seen through their mutual interface (the information flows between the dynamic object and the Terminator Object).

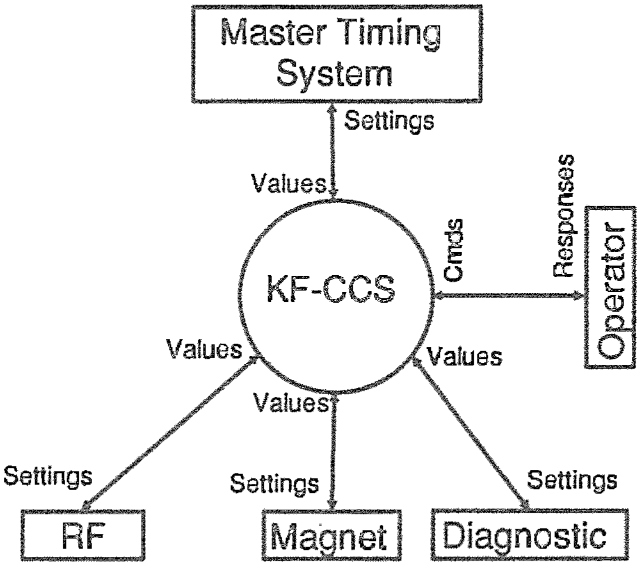


Fig. 1 A simplified Context Diagram of the KF-CCS. The single bubble represents the system-to-be-built; boxes represent Terminator Objects.

During the KAON Factory Study the premature focussing on KF-CCS systems-analysis [3] led to some difficulties with users and reviewers appreciating the role of the (predominantly software based) KF-CCS in the much larger context of controlling KAON Factory beam production, and the affect of interactions between its Terminator Objects. A Super-Context Diagram was therefore created to show the "bigger picture" (Fig. 3). In the KAON Factory, interactions between Terminator Objects will be due, for example, to the Master Timing System that will determine (predominantly in hardware) the exact timing of all beam related events e.g. beam transfers between any of the 5 rings.

Domain driven modelling [2], on the other hand, does not *require* this premature move into systems-analysis. In the early phases of KF-CCS planning a more general form of modelling would have been extremely helpful in understanding the operation of the KAON Factory. With this knowledge in hand, one or more systems analyses could have been performed to ascertain the suitability of automating various tasks as part of the KF-CCS.

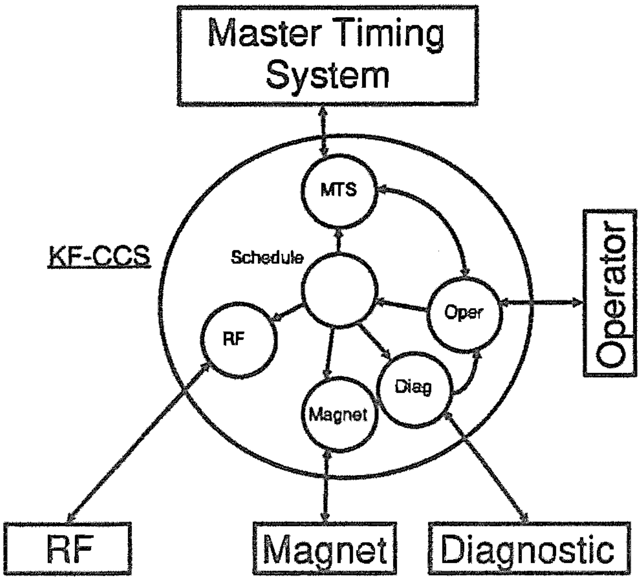


Fig. 2 The Object Communication Diagram for Fig. 1 (inside the bubble) represents the composition of the KF-CCS in terms of system-objects and the information flows between them and the external Terminator Objects.

Models that allow the precise description of *what* is required to be accomplished in a particular field of interest and *when*, but regardless of *how* it is to be carried out, are termed domain models. A complete KAON Factory domain model would show the desired behavior of the KAON Factory regardless of whether control systems had been constructed to achieve the behavior or whether the devices naturally embodied the required behavior. To precisely define the nature of domain models and their use in control system building, a brief introduction to domains follows.

III. DOMAINS

One way humans organize their knowledge of the world is in terms of domains e.g. the domains of banking, physics, art, law, etc. A domain serves as a context within which technical terms and expressions usually have a single meaning; for instance, the expression "*The grounds are fine.*" has completely different meanings in the domains of electronics, gardening, debating and coffee making. The term "domain specific language" highlights this re-use of old words, with new meanings, as an appropriate manner of describing a domain.

Domains can include other domains (e.g. divorce laws are part of all laws), can overlap other domains or be completely independent of each other. The contents of a domain are determined by a set of criteria termed Domain Criteria. It is implicitly understood that the criteria for common-place domains are known to most people. Thus, for example, no-one would consider "*Magnets*" as being part of "*Banking*".

We can capture the essential details of a domain by making models. These models take many different forms, for example:

- written text, for qualitative domains like medicine, or
- written text with embedded mathematical formulae, for quantitative domains like physics.

These models usually describe a domain in terms of five important types of knowledge, namely:

- concepts used by experts working within that domain,
- facts about the individual concepts,
- facts relating two or more concepts,
- interactions that occur, and
- events and conditions that cause interactions within the domain.

This time honoured approach to describing domains, in terms of the "things" that experts believe are part of the domain and the relationships between them, has recently been called "the object-oriented approach" by designers and builders of software systems. The object-oriented approach has been gathering substantial support from vendors and builders alike in recent years due to its uniform manner of modelling both problems and solutions, and the availability of languages that support "software objects". The support has been a "grass roots" movement, arising initially from an understanding of the benefits accrued by using object-oriented programming languages on a project; and later expanding to encompass the earlier analysis and design modelling phases of a software project.

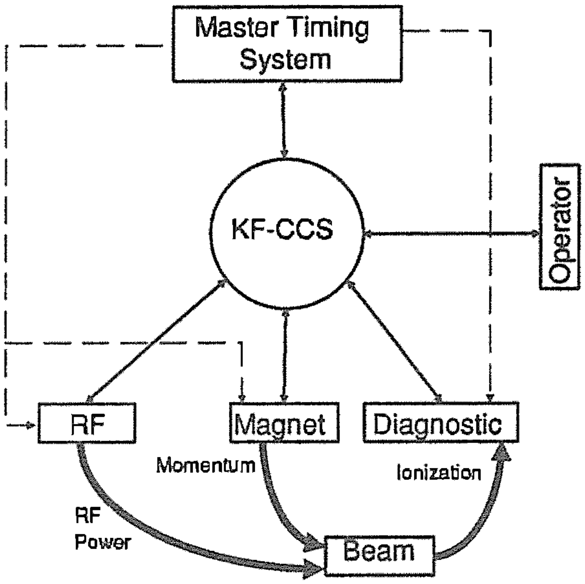


Fig. 3 The Super-Context Diagram of Fig.1 illustrates the "fit" of the KF-CCS into its immediate environment.

IV. DOMAIN BASED MODELLING

One can "view" domain based modelling as being one more step down the historical path of increasingly structuring systems

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S1400P02

during their construction. Originally, coding was performed in machine code; later in assembler and today in FORTRAN-like languages; structured programming, structured design, objects and applications were more recently created to manage the complexity of large software systems.

A more general understanding of this structuring can be obtained by "viewing" it as successful attempts at constructing new domains with associated domain specific languages. The elements of each new domain were abstractions of useful features from older domains; thus Structured Design highlighted the composition of *programs* from *procedures calling* other *procedures* and *passing data/control couples*. Its approach was language independent in that it abstracted away the details of any procedure's implementation but maintained the usefulness of being able to consider groups of statements as producing definite results from given inputs.

All traditional and most contemporary, commercially available object-oriented development methodologies and languages are presently not domain based in that they do not recognise the existence of domains and their Domain Criteria. In solving a particular problem it is, therefore, possible to incorporate any object what-so-ever into the solution domain. It would strike most of us as being unusual if a "Banking" system specification required "Magnets" but it does not seem unlikely that a "Control System" specification should require "VME".

Both of these examples violate the basic structuring tenet of any domain driven methodology that attempts to prevent the mixing of elements of a given problem (the problem domain) with elements of a particular solution (the solution domain).

Most control systems builders support this domain-driven development heuristic but express it in terms of "layered" designs for control systems - a layer's contents, ideally, being modifiable without affecting adjacent layers [6,7]. In such "good" designs, elements from one layer do not appear in other layers e.g. *network links* inside the *user interface layer*.

Modelling in a scientific domain has often led to a formal mathematical representation of the relationships between the important concepts in the domain but to a completely informal treatment of the 4 other types of knowledge. Physics is a prime example of this modelling approach.

The object-oriented approach to modelling in a domain seeks to redress this imbalance and reduce the emphasis on relationships. To achieve this goal, object-oriented domain analysis embodies a number of model-types that are unfamiliar to physicists and to most builders of automated systems. The models-types that capture the five types of knowledge, cited above, are:

- Extended Entity Relationship Diagrams (EERD); to model the entities and concepts in a domain, their properties and the relationships between them,
- State Transition Diagrams (STD); to model the modes of behavior of each entity and the causes of transitions between the modes,
- Object Structure Diagrams (OSD); to model the processes inherent in each entity and how these effect entity properties and behavior,
- Object Interaction Diagrams (OID); to model the causal relationships between the entities.

Domain models are, therefore, models of an entire field of knowledge in the same way that "PV=nRT" is a physicist's model of an Ideal Gas (a model which could be incorporated into an object-oriented domain model of Ideal Gases as "viewed" by physicists).

Domain models can only be validated by:

- experimenting with physical entities in the domain (e.g. RF cavities and Magnets for KAON), or
- questioning domain-experts about conceptual entities in the domain (e.g. a Beam Schedule, Startup Sequence),

and comparing the results with predictions from the domain models. In like manner the completeness of domain models can only be established by consulting a domain expert; but once the domain models are deemed complete and valid they can serve as a re-usable resource for projects undertaking the automation of activities in the domain. Domain models serve as the *only* criteria against which the "correctness" of any automation project is established.

V. MODELLING IN A DOMAIN

To manage the complexity of modelling in a given domain an observer should represent the domain from a particular viewpoint. Those features that the observer deems *essential* to the viewpoint must be included in the models while all other irrelevant features are omitted. The resulting model is a particular abstract view of the domain. The most important viewpoints of a domain, termed the Canonical Domain Views, are those of the different types of people working in the domain.

The domains relevant to the builders of the KF-CCS were most easily identified from an analysis of all personnel that will be involved in the production of the 30 GeV proton beams [3]. As personnel are frequently assigned multiple jobs, the study focussed on eliciting the roles to be played by those personnel in producing beam. The view of the KAON Factory perceived by any person acting out a particular role defines a description of some relevant (canonical) domain. As several roles often share a common or similar view of the KAON Factory it follows that the number of Canonical Domain Views is limited by the number of roles identified.

An example KAON Factory beam-delivery domain-model (Fig. 4) clearly shows the Master Timing System (MTS), its interaction with each beamline device and with an Operator. Personnel assuming the latter role use the MTS to tune the beam in the synchrotrons after all beamline devices have been turned on. Unlike in Fig. 1, this simple domain model does not distinguish between those functions carried out by hardware or software, and corresponds to the way Beam Physicists describe the operation of the KAON Factory.

A complete model of the domain, formed from a union of the Canonical Domain Views, is termed the Canonical Domain Model.

VI. CANONICAL DOMAINS RELEVANT TO CONTROL SYSTEM BUILDERS

In the KAON Factory domain, which is concerned with delivering a 100 microampere proton beam to 30 GeV, the different types of personnel involved [3] have been used to identify several canonical domains, for example:

- the beam-delivery domain; in which KAON Factory Operators and Beam Physicists concern themselves with producing an optimum 30 GeV proton beam and gaining an understanding of the synchrotrons,
- the equipment management domain; in which Equipment Specialists concern themselves with monitoring and controlling devices required to deliver the proton beam,
- the KF-CCS equipment management domain; in which KF-CCS electronics technicians concern themselves with maintaining the equipment related to the KF-CCS operation,
- the KF-CCS implementation domain; in which KF-CCS Managers, Analysts, Designers and Programmers concern themselves with the construction of software according to specifications deduced from the above domains.

Clearly there are relationships between these domains. The last 2 domains are solution domains for problems arising in the first 2 domains, but not vica versa; that is, in trying to implement the required behavior of KAON Factory beam transport devices, when delivering beam, it is easiest to employ ***Intelligent*** control system technology to control relatively ***un-intelligent***, passive beam transport devices.

We noted earlier that the requirements for the KF-CCS could have been derived by performing a systems analysis on a relevant KAON Factory domain model, had it been available. Similarly, the requirements for the KF-CCS ***implementation*** could be derived by performing:

- a domain analysis of control systems implementation technologies, followed by
- a system analysis of the domain to highlight the separation of control system functions between available hardware and software.

The latter analysis was performed during the 1989 KAON Factory study for hardware costing purposes, while the former analysis is presently underway.

VII. CONTROL SYSTEMS IMPLEMENTATION DOMAIN

Control systems experts are well aware of their domain of expertise. Figure 5 shows an (incomplete) EERD of a traditional, generic central controls system implementation domain, highlighting the major entities in the domain and some of the relationships between them. A short explanation of the EERD follows with the entities capitalized for ease of reference the first time they are referred to.

All working DESIGNERs and PROGRAMMERs use WORKSTATIONs to create the source classes from the CONTROL SYSTEM SPECIFICATION. The implementation language, represented by COMPILER, is object-oriented, supporting the separate compilation of SOURCE CLASSes that are then linked into an EXECUTABLE IMAGE by the LINKER. The executable images are run within TASKs by a CPU in a PROCESSOR MODULE. They communicate with each using ITCs (inter-task communication links), if on the same processor, and LAN LINKs if on different processors.

Each beam transport and beam diagnostic DEVICE is interfaced to the processors by a single DEVICE I/O MODULE placed in a VME CRATE

Processors have 2 types of storage: CPU MEMORY inside the processor and disks mounted in DISK MODULEs. The former provides volatile storage that is lost whenever power fails, the processor fails, the processor is removed etc., while the latter provides persistent storage that survives such an occurrences.

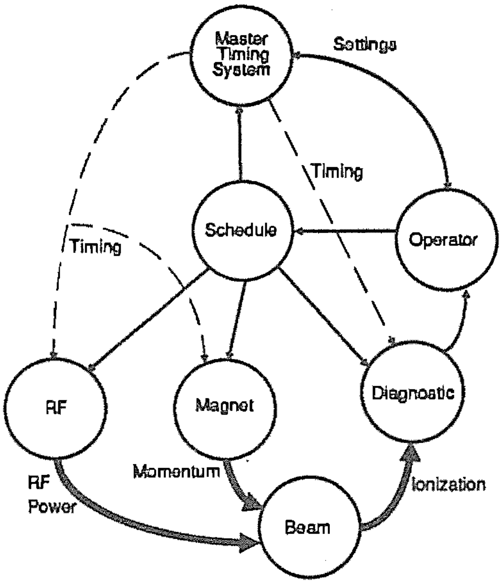


Fig. 4 The Object Interaction Diagram for Fig. 1. This domain model shows the entities in the domain as circles. Heavy, directed lines are energy flows while thin-solid and dashed lines are information flows.

VIII. MAPPING

The CONTROL SYSTEM SPECIFICATION entity (top right of Fig. 5) is a model of a control system to be built. To simplify discussions we will postulate this to be fully described by only one model type, namely an Object Communication Diagram (OCD); Fig. 2 for example. The OCD is, in turn, composed of:

- a set of SYSTEM-OBJECT MODELs,
- a set of SYSTEM TERMINATOR MODELs, and
- a set of all INTER-OBJECT FLOW MODELs describing information flows between System Objects and between System Objects and Terminator Objects.

In addition, each SYSTEM-OBJECT MODEL describes:

- the INTRA-OBJECT FLOWs accepted by the object,
- the PROCESSes required to convert input flows to output flows and

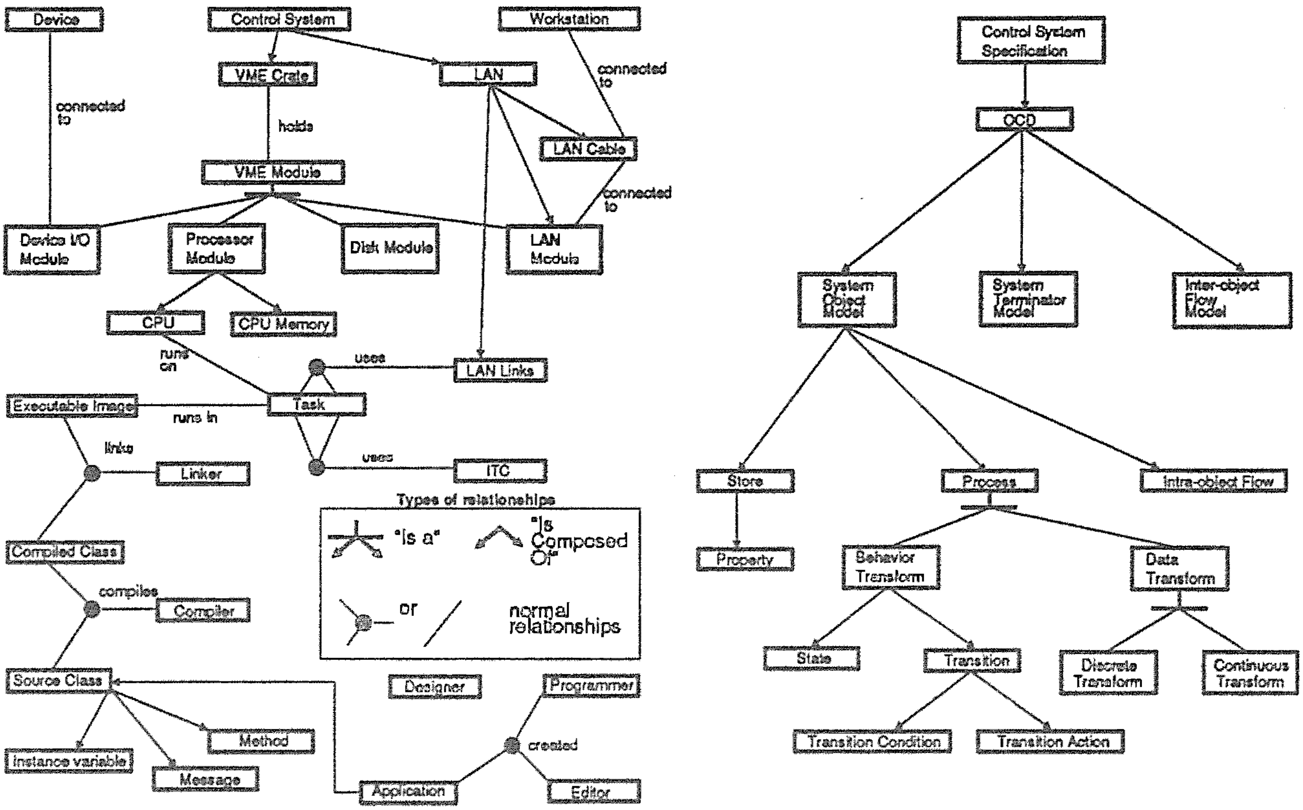


Fig. 5 An (incomplete) Extended Entity Relationship Diagram of a traditional controls system implementation domain. Entities in the domain are represented by boxes. A legend describes the type of relationships between entities.

- the information needed be STOREd to satisfy the PROCESSes.

Each of the above models is a description of a particular type of *need* that entities in the implementation domain must fulfil. That is, we must find entities in the implementation domain that are capable of performing as required by the models. This association of a model with an implementation entity is termed *mapping*. Performing a mapping (i.e. translating) is similar to compiling the description of an algorithm, in a high level language, into an equivalent description in an assembler language. The latter could , in turn, be assembled, linked and executed.

To illustrate the notion of mapping we will consider the various ways of implementing a CONTROL SYSTEM SPECIFICATION using our suite of implementation entities (Fig. 5).

When all system-object PROCESSes are of the discrete variety and we are using a traditional processor, we can map (denoted by the "-->" symbol) these models as follows:

- both types of "flows" --> the MESSAGEs accepted by a SOURCE CLASS,
- "processes" --> the METHODs of a SOURCE CLASS (a named-method is invoked by the class receiving a message with the same name), and
- "stores" --> the INSTANCE VARIABLES of a SOURCE CLASS,
- "system-object model" --> SOURCE CLASS,
- "control system specification" --> APPLICATION, and
- "terminators" --> DEVICE I/O MODULEs.

This mapping implies:

- the control system software is a single application program,
- that it deals with events in the real world sequentially, and
- that all terminator I/O modules must be in the same VME crate.

In a large control system this latter implication is unacceptable. The distributed nature and quantity of terminators will require a large number of terminator I/O modules in several VME crates to interface them to the application software. A local processor will also be required in each crate to access the I/O modules in that crate. The relevant application software must also be running in that processor.

From the OCD of Fig. 2 one can see that the simplest choice of application software to be run in a particular processor must correspond to the system-objects for the terminators interfaced into that crate. Flows between each terminator and the processor would then occur within the single crate.

With this choice of allocating system-objects to processors the original single program now becomes fractured into several programs. Now flows between these distributed system-objects cannot all be mapped to messages; those between objects on different processors must be mapped to LAN LINKS.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S1400P02

Now consider the changes required when some PROCESSes are CONTINUOUS TRANSFORMs. As these processes must be able to proceed independently they must be mapped to TASKs. In addition the INTRA-OBJECT FLOWs to these processes must be mapped to ITCs (inter-task communication channels). Now a single system-object may be distributed across several tasks, and there is no reason for these tasks to reside on a single processor. If a system-object is fractured across processors then its INTER- and INTRA-OBJECT FLOWS must be mapped to ITCs and LAN LINKs.

IX. PERFORMING AND EVALUATING THE MAP

Given an application program, implemented as a set of traditional communicating programs running on a set of processors, there are many reasons for selecting a particular arrangement of programs to processors. Criteria such as:

- access to critical resources e.g. particular I/O modules,
- communications volume to and from critical resources,
- response times to events,
- meeting deadlines (in hard real-time systems),
- computational "volume" per mode of operation,
- computational "volume" per event,
- processor's capability, capacity and reliability,
- storage capability, capacity and reliability,
- network capability, capacity and reliability,

play important roles in determining the positioning of programs, regardless of operating system and language issues.

The mapping process, described above, leads ultimately to the allocation of parts of SYSTEM-OBJECTs to multiple (sub-application) programs. To simplify the establishment of this mapping, initially, we have studied the case in which:

- system-objects were not fractured smaller than individual PROCESSes, and
- all PROCESSes communicating with a terminator via its I/O module were allocated to processors in the same crate.

The number of application program fragments to distribute is then large but finite. We are presently investigating an algorithm that would produce a "best" arrangement of programs among processors by minimizing a "quality factor". The "quality factor" depends on the values of all of the properties listed above. These values will all be available from a control system implementation-domain model presently being completed for the KAON Factory.

The simple mapping model presented here has difficulties accounting for certain properties required by some systems - for instance "fault tolerance". In a fault-tolerant system one would require copies of a program to execute on different processors - a feature that would increase the amount communications over a solution in which all copies were allocated to the same processor. The role of a processors "reliability" property in the "quality factor" must be, therefore, to "repel" copies of like software.

X. CONCLUSIONS

The possibility of generating a mapping directly from a control system specification now seems feasible. It relies upon:

- detailed domain and system models of the control system implementation architecture (e.g. Fig. 5) and either
- a detailed mapping scheme that captures the experience of control system designers, or
- a calculable figure of merit associated with each mapping that could be minimized, say, to produce the "best" design.

Commercial code generator products have been available for several years that are capable of creating executable systems directly from structured designs. They are feasible because the commercial computing domain is very mature and code generators tend to work with mainstream transaction management products for which all interface software is provided.

It is timely to pursue the goal of developing control systems at a higher "level" of abstraction. There are many benefits to be gained by leaving the "technical details" to a new form of "compiler" in much the same way as assembler programs were replaced by high level language programs. The final products will be more uniform in design and more adaptable to technology changes by "re-compilation". Future control system development work at TRIUMF in support of the KAON Factory will be directed toward developing such a "control system compiler".

XI. REFERENCES

[1] D.A. Dohan, G.A. Ludgate, E.A. Osberg, S. Koscielniak and C. Inwood, "Definition study of the TRIUMF KAON Factory control system project", Proceedings ICALEPCS '89, Nucl. Instr. and Meth. A293 (1990) 6-11.

[2] C. Inwood, G.A. Ludgate, D.A. Dohan, E.A. Osberg and S. Koscielniak, "Domain-driven specification techniques simplify the analysis of requirements for the KAON Factory central control system", Proceedings ICALEPCS '89, Nucl. Instr. and Meth. A293 (1990) 390-393.

[3] E.A. Osberg, G.A. Ludgate, S. Koscielniak, D.A. Dohan and C. Inwood, "Dynamic object modelling as applied to the KAON control system", Proceedings ICALEPCS '89, Nucl. Instr. and Meth. A293 (1990) 394-401.

[4] J-L. Theron, "Design and checking of a large Ada real-time system", Proceedings ICALEPCS '89, Nucl. Instr. and Meth. A293 (1990) 373-376.

[5] G. Morpurgo, "SASD and the CERN Run-time coordinator", Proceedings ICALEPCS '89, Nucl. Instr. and Meth. A293 (1990) 385-389.

[6] B. Kuiper, "Accelerator controls at CERN: some converging trends", Proceedings ICALEPCS '89, Nucl. Instr. and Meth. A293 (1990) 308-315.

[7] P. Clout and A Daneels, "Report of the Los Alamos accelerator automation application toolkit workshop", Proceedings ICALEPCS '89, Nucl. Instr. and Meth. A293 (1990) 321-324.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S1400P03

The Direct Manipulation Shell: Creating Extensible Display Page Editors

Michael E. Allen, Michael Christiansen
SSC Laboratory *
2550 Beckleymeade Ave.
Dallas, Texas 75237

Abstract

Accelerator controls systems provide parameter display pages which allow the operator to monitor and manipulate selected control points in the system. Display pages are generally implemented as either hand-crafted, purpose-built programs; or by using a specialized display page layout tool. These two methods of display page development exhibit the classic trade-off between functionality vs. ease of implementation. In the Direct Manipulation Shell we approach the process of developing a display page in a manifestly object-oriented manner. This is done by providing a general framework for interactively instantiating and manipulating display objects.

I. INTRODUCTION

We are developing a tool, known as the Direct Manipulation Shell (DMS), which will allow the construction of software applications in much the same way as modern hardware devices are constructed. That is, the user selects and combines together software components in an interactive, plug-and-play fashion when developing their applications software. DMS provides an environment in which software components are provided and directly manipulated (hence Direct Manipulation Shell) by the user.

A programming environment developed through DMS contains software components which address the needs of a single problem domain, e.g., accelerator parameter page development. This can be contrasted to a traditional programming environment containing compilers, linkers, editors, etc., which support no specific problem domain and provide no domain specific support for applications development. Using DMS, the user performing the applications programming spends most of his time browsing catalogs of domain specific components rather than developing algorithms and data structures. It is assumed that this applications programmer is knowledgeable in the domain supported by the specific environment, not necessarily in the domain of the computer sciences.

The goal of DMS is to provide users with a software development environment in which they construct solutions in problem domains about which they are concerned and knowledgeable. These domain experts are provided components that are presented and manipulated through terms and concepts found in this problem domain. Using the facilities provided by DMS, programming experts develop a set of interrelated software components which can be used to construct solutions to problems in this domain. This process of constructing a programming environment through DMS is similar to the process of constructing an expert system [2] using an expert systems development shell. When developing an expert system, a team of programmers and domain experts combine efforts to develop a set of rules which address problems in a specific problem domain. With DMS, a team of programming and problem domain experts construct a set of software components which can be accessed and manipulated through DMS. In either case the user is provided with an environment which can be applied to problem solving with little understanding of the underlying computing environment.

Of course, theses goals are not unique to DMS. Basically, DMS provides an interactive, interpreted, object-oriented, programming environment. Usually, such environments have the following major shortcomings:

1. performance.
2. availability of third party, off-the-shelf "components".
3. performance.

For an accelerator control application (1) and (3) (and to a lesser extent (2)) can be killers. The DMS environment is designed to specifically address these limitations. This is achieved in the current version of the DMS tool via the use of a modified Common Lisp [5] interpreter, XLisp [1].

XLisp has incorporate within it an object-oriented language constructs which allow classes to be defined and instances of XLisp object to be created. Our modifications to XLisp enable the user to interactively create and manipulate instances of XLisp objects which in turn create and manipulate instances of C++ objects. This is much more than just a foreign function interface, because the objects thus created are now managed by the DMS environment. This means, for example, that much of the memory management is taken care of automatically (garbage collection). Additionally, DMS knows about C++ data structures, so that unmodified C++ code can be linked directly into the DMS environment. Additionally, because of the object oriented extensions on the Lisp side, one can write straight forward Lisp code without continually bothering about how data is represented.

Within DMS one can move freely between the Lisp and C++ environments, taking advantage of the best features of both. In particular, one may take advantage of the speed and availability of C++ class libraries within an interactive Lisp programming environment. We have, for example,

---

*Operated by the Universities Research Association, Inc., for the U.S. Department of Energy under Contract No. DE-AC02-89ER40486.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S1400P03

incorporated both the GNU [7] and InterViews [8] class libraries into a development environment for user interface development. The process of incorporating, or linking in, new C++ classes is discussed in some detail in a later section. It is important to note, however, that no modifications to the C++ code is required. In fact, one does not even need access to the source code in order to integrate a C++ class library into the DMS environment.

Another reason for integrating the Lisp programming language into the DMS tools is the opportunity to apply expert system, logic programming, and other knowledge based technologies in the development of domain specific development environments. Not only does the DMS tool support the interactive construction of software solutions, but embedded rules and constraints systems can guide the software developer in the correct manipulation and combination of software components into a needed solution.

II. THE C++-XLISP INTERFACE

The connection between the XLisp and C++ programming environments is accomplished by providing an interface between the Lisp and C++ run-time environments. In general, a method call on a XLisp object is translated into a call on a C++ object's method. This translation is accomplished by a C++ function, called the Interface Function, which is generated specifically for the purpose of providing an interface between the Lisp and C++ environments.

Each C++ class which is imported into the XLisp environment is interfaced to DMS through an XLisp class. An XLisp class which imports and makes available a C++ class is called an "import class". Each import class duplicates the set of methods that the C++ class provides. When an instance of an import class is created, the constructor method for the import class constructs and makes available an instance of its corresponding C++ class. An instance of an import class is called an "interface object" and each interface object maintains and provides an interface to a single instance of a C++ object.

An Interface Function is created specifically for each class/method combination and is responsible for translating the Lisp arguments provided to the XLisp method call into C++ arguments which are passed to the C++ method call. The Interface Function also translates the value returned by the C++ method into a Lisp variable which is returned as the result of the XLisp method call on the interface object.

When an instance of a XLisp Interface Object is created through the interaction of the user with the XLisp programming environment, an XLisp constructor method calls an "interface function" which creates an instance of the imported C++ class. The pointer to this new C++ object is returned by the interface function and assigned to a pointer instance variable maintained by the interface object. This C++ pointer is then used as a target for all future interaction which occurs between the interface object and the C++ object it maintains.

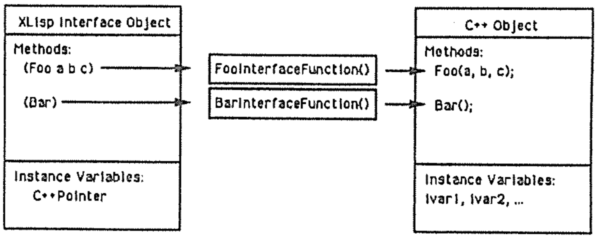


Figure 1: Interface between XLisp import object and C++ object.

The C++ pointer maintained by the interface Object to its C++ object instance can be manipulated and passed as an argument in other XLisp method calls. Thus, the address, or location, of a C++ object can be passed to other C++ objects. In this way direct interaction between C++ objects can established once references between these C++ objects have been passed through their interface objects. The importance of this direct interaction lies it the fact that once established, directly interacting C++ objects can execute method calls with the speed of compile code in a traditional, statically linked, programming environment. Solutions to problems (application programs) are then composed utilizing C++ objects which have been created and manipulated through their Interface Object instances.

III. ADDING INTERFACES FOR NEW C++-CLASSES TO DMS

Integrating a new C++ class into the DMS image refers to the process of integrating the class data structure and its methods into the DMS process (or image). This integration can be accomplished either statically or dynamically. Static integration is implemented by simply linking the compiled object code which implements the class methods, data structures, and interface functions into the DMS image at link time. Dynamic integration is accomplished using a public domain library called Dld [4,3] which provides the ability to dynamically load and relocate object code into an executing image at run-time. * The Interface Functions needed for each method provided by an interface object are generated automatically using the development environment provided by the DMS tool.

The DMS development environment described in the above figure provides the ability to integrate new C++ classes into a DMS supported programming environment. The user wishing to integrate a new C++ class provides a description of the class and its methods in a form very similar to the typical C++ header declaration. This description is parsed for errors and is translated into a XLisp class declaration and a set of interface functions for each method in the C++ description.

---

*Dynamic integration is currently only offered in the Sun OS environment.

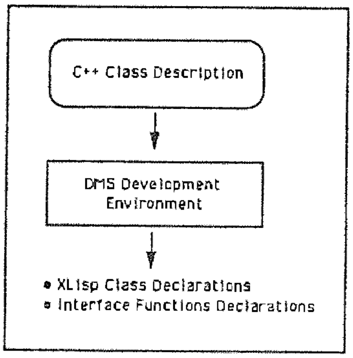


Figure 2: Incorporating C++ classes into the DMS environment.

In the case of the statically integrated DMS tool, the interface function declarations are compiled using the native C++ code development tools into an object code format. This object code is linked with the DMS object code, along with the C++ methods to generate the desired DMS programming environment. Also generated from the C++ class description are a set of XLisp class declaration which implement the import classes described above. These XLisp class declarations are loaded into a DMS environment when the tool is started.

IV. CONCERNS AND FUTURE DIRECTIONS

As noted, DMS is currently implemented as an extension to the XLisp programming environment. DMS runs on on a variety of Unix workstations, and requires the X-Window system to support its user interface. Also integrated with the tool is a text editor which allows the development and execution of XLisp programs in a mouse driven editing environment. Within the editor the user is able to write XLisp source code with such capabilities as parenthesis matching, multiple widows, and text select, cut, and paste operations supported through the mouse interface. XLisp source code can be interactively executed directly from within the editor by selecting and evaluating the code of interest through the mouse interface. This feature is similar to Smalltalk's Workspace object, and allows work sessions to be saved and restored.

The primary programming environment to have been developed is an X-Window user interface development environment based on the InterViews [8] C++ class library and our extensions to that library, called glistk [6]. Using this environment the user is able to interactively create and exercise user interfaces implemented as InterViews and glistk objects. In addition, glistk's provide an underlying inter-object communication mechanism that has also been extended into the XLisp environment. Furthermore, there are also XLisp PushButtons and other Lisp based interactive objects. These Lisp based objects allow the development of higher level functionality within the XLisp programming environment. In particular, general Lisp expressions can be executed when these objects are selected.

There are two main concerns about the current DMS environment. The first is that XLisp, while providing an object-oriented interface to Lisp, is not standard. We would like to move DMS to a CLOS and Common Lisp environment. This would make accessible the rich and robust environments and tools available with commercial Lisp implementations. We are currently evaluating a few different options.

The second concern is actually more serious. Developing and maintaining a DMS programming environment is a reasonably complex process. When several dozen C++ classes are to be integrated into an environment, house keeping and version control become complex and error prone. Further, developing a new environment sometimes requires a in-depth understanding of XLisp internals. In the current version, integrating a new class involves several processing stages which could be combined into a few simpler steps.

This situation could be improved in a couple of ways. First, it should be possible to generate import classes and interface functions automatically from C++ header files. Another future enhancement would be to eliminate the need to generate and link Interface Functions for each method provided by an interface object. This might be accomplished using a byte code interpreter which interprets a set of byte codes describing the types of arguments expected for a method call and which uses these codes to translate Lisp to C++ arguments and then performs the C++ method call.

References

[1] D. M. Bentz. XLisp: An object-oriented lisp. version 2.0. Available as public domain software., 1988.

[2] F. Hayes-Roth, D. Waterman, and D. Lenat. *Building Expert Systems*. Addison-Wesley, Reading, Massachusetts, 1983.

[3] W. Wilson Ho. An Approach to Genuine Dynamic Linking. Available as Technical Report.

[4] W. Wilson Ho. Dld: A Dynamic Link/Unlink Editor, Version 3.2.3. Available as technical report.

[5] G. L. Steele Jr. *Common Lisp: The Language*. Digital Press, Bedford, Massachusetts, 1984.

[6] M. Kane, C. Saltmarsh, V. Paxson, M. Allen, and P. Veals. The GLISTK Manual. Available as internal technical memorandum.

[7] D. Lea. The GNU C++ Library. Available from the Free Software Foundation.

[8] M. A. Linton, J. M. Vlissedes, and P.R.Calder. Composing User Interfaces with InterViews. *IEEE Computer*, 22(2), Feb. 1988.

Object Oriented Programming Techniques Applied to Device Access and Control

A.Götz, W.D.Klotz, J.Meyer
ESRF, BP 220
F-38043 Grenoble Cedex
FRANCE

1 Introduction

Device access and device control is one of the most important tasks of any control system. This is because control implies obtaining information about the physical world by reading sensors and modifying the behaviour of the physical world by sending commands to actuators. At the **European Synchrotron Radiation Facility (ESRF**, ref. [1]) effort has gone into designing and implementing a model for device access and control using as much as possible the latest ideas and methods of Software Engineering. One of the main contributions in recent years to Software Engineering has been in the field of **Object Oriented Programming(OOP)**. Although the philosophy is not new the refinement and application of this methodology on a wide scale is. At the ESRF a model for device access and control has been developed which is based on **OOP** methods. This model, called the **device server model**, is the topic of this paper. The device server model is written entirely in C and is therefore portable. It depends on no other software and can be ported to any machine where there is a C compiler. Because the model is based on **OOP** it presents a *user-oriented* view of the world as opposed to a *software-* or *hardware-oriented* view of the world.

This paper will describe the device server model. It will describe the problem of device access and the advantages of using **OOP** techniques to solve it. It will present the model. The methodology used to implement **OOP** in the device server model called **Objects In C (OIC)** will be described. An example of a typical device server at the ESRF will be presented. The experience gained from the device server model will be discussed. The paper will conclude with a discussion on how the device server model could be standardised to treat a wider range of problems.

2 The Device Access Problem

The problem which the device server model is designed to solve is a problem which every control system is faced with. The problem could be described as – *how to provide access and control for all the physical devices which represent the machine ?*

Unfortunately there is no widely accepted industrial standard for interfacing devices to computers. Although some attempts have been made at defining an industrial standard none of them have succeeded (see [2]). This means that there are about as many ways to interface a device to a computer as there are device suppliers.

The device access problem would be simple if a single standard were adopted for interfacing. In reality however this turns out to be too expensive because it involves extra development costs for the suppliers.

3 The Device Server Model

At the ESRF a unified model (called the **device server model**) has been developed to solve the problem of device access and control. It is unified for two reasons —

- it presents a single interface for upper level applications to all kinds of devices, and
- it defines the framework within which to implement device access and control for all devices.

The model can be divided into a number of basic elements - the *device*, the *server*, the *Objects In C* methodology, the *root class*, the *device class*, *resource database*, *commands*, *local access*, *network access*, and the *application programmers interface*.

3.1 The Model

The basic idea of the device server model is to treat each **device** as an **object** which is created and stored in a process called a **server**.

Each device is a separate entity which has its own *data* and *behaviour*. Each device has a *unique name* which identifies it in network name space. Devices are configured via **resources** which are stored in a **database**. Devices are organised according to classes, each device belonging to a class. Classes are implemented in C using a technique called **Objects In C**. All classes are derived from one **root** class. The class contains a generic description of the device i.e. what actions can be performed on the device and how it responds to them. The actions are made available via commands. Commands can be executed *locally* or *remotely* (i.e. across a network). Network access to a device and its commands is provided by an application programmers interface using a remote procedure call.

3.2 The Device

The device is at the heart of the device server model. It represents a level of abstraction which previously did not exist. A device can be a physical piece of hardware (e.g. an interlock bit), an *ensemble* of hardware (e.g. a screen attached to a stepper motor), a logical device (e.g. a taper), or a combination of all these (e.g. a storage ring). Each device has a unique name. At the ESRF a three field name space (consisting of DOMAIN/FAMILY/MEMBER) has been adopted.

The decision of what level of abstraction a device represents depends on the requirements of the clients. At the ESRF these are the machine physicists. Devices should model the clients view of the problem as closely as possible. Hardware dependencies should be hidden behind the device abstraction. For example if a corrector consists of three independant powersupplies the client (assuming she is a machine physicist) should only see a single device — the corrector, and not three independant devices.

All devices are treated as having state. Each device has a list of commands which it understands. Before any command can be executed the state machine gets checked to see if the command can be executed in the present state. The commands and the state machine are implemented in the device's class.

3.3 The Server

Another integral part of the device server model is the server concept. The server is a process whose main task is to offer one or many *service(s)* for *clients* who want to take advantage thereof. The server spends most of its time in a *wait-loop* waiting for clients to demand its service(s). This division of labour is known as the client-server concept. It exists in different flavours and is very common in modern operating systems.

The adoption of this concept in the device server model has certain implications. The server in the device server model is there to serve one or many device(s) i.e. there is one server per device but there can be many devices per server. The exact configuration depends on the hardware organisation, CPU charge, and the available memory. The fact that there can be many devices per server means that a single device should not monopolise the server for more than a pre-defined amount of time. Otherwise the server is blocked and new or existing clients will not be able to connect to the same or other devices served by that server. The server waits for clients by listening at a certain network address. The mechanism for doing this is implemented by a remote procedure call. At the ESRF the network addresses are determined dynamically (at server startup time) and then stored in a *database*. The first time a client connects to a server it goes to the database to retrieve the server's address, after which it communicates directly with the server.

3.4 Objects In C

The use of objects and classes in the device server model necessitates appropriate OOP tools. The natural choice would have been to use one of the many OOP languages which are available on the market today. If possible one for which a standard exists or will exist (e.g. C++). The choice of a language is not independant of the development evironment however. The language chosen has to be fully compatible with the development environment. At the ESRF the device access and control development environment consists of OS9, HP-UX, SunOS and the SUN NFS/RPC. Unfortunately there is no commercially available OOP language compatible with this environment. The only language which supports the above environment is C. In order to use OOP techniques it was therefore necessary to develop a methodology in C, called Objects In C (from here on OIC). The methodology developed is implemented entirely in C and is closely modelled on the widget programming model (ref. [3]).

OIC implements each class as a structure in C. Class hierarchies are supported by subdividing a class structure into *partial* structures. Each partial structure representing a super- or a sub-class. Each class requires a minimum of three files :

- a private include file describing the class and object structures,
- a public include file defining the class and object types as pointers to structures and the class as an external pointer to the class structure,
- a source code file which contains the code implementing the class.

The private include file is used to define constants and/or variables which should not be visible to the outside world (the inverse being true for the public include file). All functions implementing the class are defined to have static scope in C. This means that they cannot be accessed directly by any other classes or applications – they are only accessible via the method–finder or as commands. This enforces code-hiding and reinforces the concept of encapsulation – a way of reducing coupling between software modules and making them immune to changes in the class implementation.

OIC implements an explicit method–finder. The method–finder is used to search for methods in a class or hierarchy of classes. The method–finder enables methods to be inherited. Two special versions of the method–finder exist for creating and for deleting objects.

Objects are also implemented as structures. Each object has a pointer to its class structure. This means that class

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S1400P04

related information (data and code) are stored only once per process for all objects of the same class.

3.5 The Root Class

All device classes are derived from the same root class, the **DevServerClass**.

The DevServerClass contains all common device server code. This includes all code related to the application programmer interface, the database connection, security, administration and so on. All device classes inherit this code automatically, this means improvements and changes are inherited too. The decision to have a single root class from which all other clases are derived has been fundamental to the success of the device server model. Other **OOP** based systems have used the same principle (e.g. the **NIH** set of classes, see [4]).

3.6 The Device Class

Organising devices into classes is an attempt to generalise on common features between devices and to hide device dependant details. The device class contains a complete description and implementation of the behaviour of all members of that class. Hardware specific details are implemented by the class in such a way that they are transparent to device server clients. Typically the first level of device classes (i.e. classes which deal directly with the first level of hardware) will use the utilities offered by the operating system (e.g. device drivers) to implement the hardware specific details.

New device classes can be constructed out of existing device classes. This way a new hierarchy of classes can be built up in a short time. Device classes can use existing devices as **sub–objects**. This means they appear as terminators in the class structure and not within the class hierarchy. This approach of reusing existing classes is classical for OOP and is one of its main advantages. It encourages code to be written (and maintained) only once.

3.7 The Resource Database

Implementing device access in classes forces the programmer to implement a generic solution. To achieve complete device independance it is necessary however to supplement device classes with a possibility to configure devices at runtime. This is achieved by the **resource database**. Resources are identified by an ASCII string. They are associated with devices via the device name. Resources are implemented in the device class. A well designed device class will define all device dependancies (e.g. hardware addresses, constants, minimum and maximum values etc. etc.) as resources. At runtime the device class will interrogate the database for the list of resources associated with each device it must serve. This is done during device creation and/or initialisation time. Resources allow device classes to be completely general and flexible.

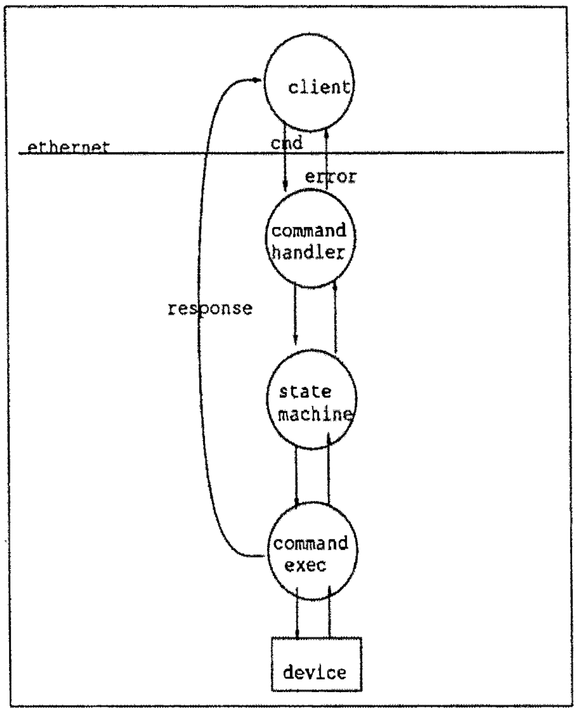


Figure 1: Command execution in the Device Server Model

At the ESRF the resources are stored offline in **Oracle** where they can be accessed using an **SQL*Forms** application. Online access is provided by a database server which interrogates a memory resident database (**RTDB**).

3.8 Commands

Each device class defines and implements a list of **commands**. The commands are the applications **dials** and **knobs** of the device. Commands (unlike methods) can not be directly inherited from superclasses. They have a fixed calling syntax – consisting of one input argument and one output argument. Arguments can be complicated structures. Commands can vary from simple **On/Off** type actions to complicated sequences which involve a large number of steps. Defining commands is the task of the class implementor and his client. The list of commands to be implemented depends on the definition of the device and the implementation of the class. Because all commands are executed synchronously care has to be taken that the execution of a command does not take longer than the maximum allowed time.

All commands are theoretically executable from a remote client across the network. It is possible however to define a restricted usage of certain (or all) commands. This ensures security for sensitive devices. Commands are executed using the application programmers interface. Before a command gets called the root class checks to see wether the argument parameters have been correctly specified. The state machine is also called to see wether the desired command can be executed in the present state. This is implemented in the **command_handler** method of the root class (see *figure 1*).

Global standard commands have been defined which are implemented in every device class e.g. **DevState** to read the device state. It is possible to define subsets of standard commands for devices belonging to the same superclass which are to be implemented in all subclass of that superclass.

3.9 Local Access

Not all clients of device access and control are network clients. It is sometimes necessary to use the device class locally. The notion of local access is completely compatible with the device server model due to the adoption of OOP techniques. Device classes can be used locally (as opposed to remotely) in a number of ways –

- as a superclass,
- as a subclass,
- as a local sub–object within a class.
- or as a local object in an application,

This allows applications which have to run close to the hardware because of performance or hardware constraints, to profit from existing device classes.

3.10 Network Access

Network access is implemented in the device server model in the root class. This is achieved by a **remote procedure call**. The *de facto* industry standard from SUN – the **NFS/RPC** has been chosen because of its wide availability.

Two types of network access are provided –

- *device*,
- *administrator*.

For this reason there are two *api's*. The device api is described below. The administrator's api supports a kind of meta-control to device servers. Using the administrator's interface various kinds of useful information can be obtained about the device server e.g. the devices being served, and the number of clients per device. The administrators interface also supports a number of commands e.g. shutting down the server, restarting the server and reconfiguring the server.

Parameters passed between clients and server have to be converted to network format (for NFS/RPC this is XDR format) and back again. These tasks are known respectively as *serialising* and *deserialising*. A central library of serialising and deserialising routines is maintained as part of the root class. This way device server programmers do not have to learn how to serialise and deserialise data.

3.11 The Application Programmers Interface

A device server client accesses devices using the application programmer's interface (*api*). In order to improve performance the device server api is based on the ***file*** paradigm. The file paradigm consists of **opening** the file, **reading** and/or **writing** to the file and then **closing** the file. The device server api paradigm consists of –

1. **importing** the device using

```
dev_import (name,ds_handle,access,error)
char       *name;
devserver  *ds_handle;
long        access;
long       *error;
```

2. **putting** and/or **getting** commands to the device using

```
dev_putget (ds_handle,cmd,argin,intype,
            argout,outtype,error)
devserver   ds_handle;
short       cmd;
DevArgument *argin;
DevType     intype;
DevArgument *argout;
DevType     outtype;
long        *error;
```

3. **freeing** the device using

```
dev_free (ds_handle,error)
devserver ds_handle;
long      *error;
```

Using these three calls a client can execute any command on a device. The client uses a local procedure call to access these functions. The local call is then converted into a network call by the *remote procedure call* mechanism (ref. [5]). Each call is a **blocking synchronous** call. This means that the client **waits** until the call returns before continuing. If the server doesn't respond or the remote machine is down a **timeout** will ocurr.

Variations of these three calls supplement the basic device server api. For example a **vector** version of the above calls exists which takes a list of devices and a list of commands to be executed, thereby reducing the network overhead incurred by the rpc. Work is also continuing on an **asynchronous** version of the **dev_putget()** call which will dispatch the command and then return immediately. The response will be queued and returned to the client when it is ready to receive it.

4 Example

The device server model has been successfully used at the ESRF to solve the problem of device access and device control for the entire **injection/extraction** part of the machine. A typical example of a device class is the **Powersupply** class.

4.1 The Powersupply

At the ESRF one of the most common device types is the **powersupply**. There are over 300 powersupplies from over 10 different suppliers. Hardware interfacing is either via a serial line or a G64 bus. Each supplier uses a different register description or protocol.

Using the device server model it has been possible to hide these hardware software differences between the different powersupplies. Applications see a generic powersupply which behaves identically for all powersupplies. The generic powersupply is a **device** in the device server model. All powersupplies belong to the same superclass – the **PowerSupplyClass**.

The PowerSupplyClass defines a partial structure for each PowerSupply device/object. Every subclass of the PowerSupplyClass uses the fields defined in the Powersupply partial structure. This way all powersupply classes have the same definitions and are easier to implement, understand and maintain.

The PowerSupplyClass is what is known as a **container** class i.e. it is never instantiated – its job is

- to provide a framework within which to implement subclasses,
- serve as a receptacle for common powersupply related methods.

Each new powersupply class is implemented as a subclass of the PowerSupplyClass. The convention has been adopted to name the powersupply classes after the supplier. *Figure 2* shows a synoptic of the powersupply class hierarchy for some of the powersupplies involved in the injection/extraction process.

In order to standardise the behaviour of all powersupplies a set of **commands** have been defined which are implemented in every powersupply class. These are –

- **DevOff**, switches the powersupply off.
- **DevOn**, switches the powersupply on.
- **DevReset**, resets the powersupply after a fault condition has ocurred.
- **DevState**, returns the state of the powersupply as a short integer.
- **DevStatus**, returns the state of the powersupply as an ASCII string.
- **DevSetValue**, sets the principal setpoint (current) to the specified value.
- **DevReadValue**, returns the last set value and the latest read value.

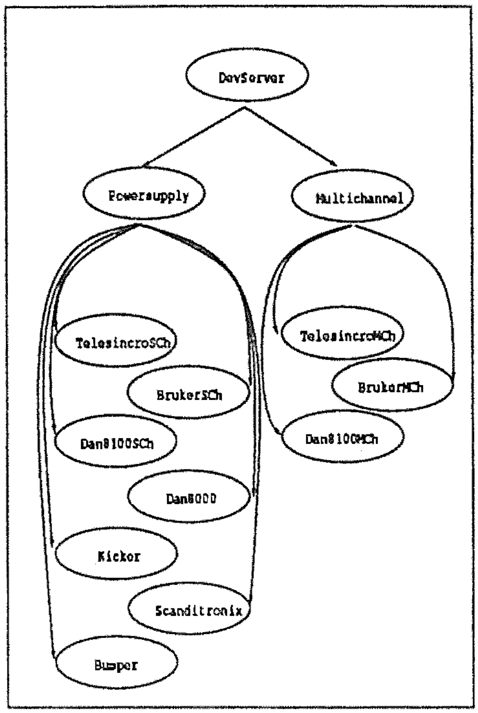


Figure 2: Class Structure for Injection/Extraction Powersupplies at the ESRF

- **DevUpdate**, returns the state, set value and read value.

For more complex powersupplies additional commands can be added to this list e.g. **DevStandby**.

The approach to define an generic device of type powersupply has proved very powerful for applications. The application programmers can develop their applications completely independantly of the device class. This way both programs can be developed simultaneously and be ready at the same time.

5 The OOP Experience

OOP techniques were chosen for the device server model partly because the problem (a general device access system) falls in the scope of problems which can be solved by **OOP** techniques; partly as an experiment in **OOP** to see what it really implies.

From the experience with the device server model the following advantages could be identified :

- the possibility to **inherit** code from existing classes,
- the **logical structure** it imposes on classes,
- the fact that it reduces the **coupling** between code to a minimum.

Not all experience with OOP has been positive however. One of the main drawbacks with OOP is the time it takes beginners to learn. Our experience has shown us that even though modelling the world in terms of objects might come naturally to many people programming with classes does not. It takes longer for programmers having no experience with OOP techniques to become productive than it takes programmers to become productive in a project using traditional (procedural-based) techniques.

6 Standardisation

Device servers can be regarded as a new generation of device drivers. They provide device independant access within a distributed computing environment. The device server model could be used as a basis for a standard way of solving device access in a distributed environment. The main areas that requires standardisation are the network protocol and the class hierarchies.

The present implementation of the device servers has defined a device access as consisting of a command with one input argument and one output argument. In order to arrive at a standard protocol the commands and the data types to be supported by the standard would need to be defined. Doing this could lead to a a standard access mechanism for devices in a distributed environment (much in the same way that the X11 protocol is a standard in graphics programming in a distributed environment).

Standardisation work is also required for class hierarchies. A standard superclass should be defined for each of the basic device types. The standard fields to be used by each member of a certain superclass will thereby be defined. A minimum set of commands to be implemented by each member of the same superclass need to be defined as well. This will ensure consistent behaviour of all devices belonging to the same superclass and maximum reusage of code.

7 Conclusion

In this paper a model, called the device server model, has been presented for solving the problem of device access and control faced by all control systems. Object Oriented Programming techniques were used to achieve a powerful yet flexible solution. The model provides a solution to the problem which hides device dependancies. It defines a software framework which has to be respected by implementors of device classes – this is very useful for developing groupware. The decision to implement remote access in the root class means that device servers can be easily integrated in a distributed control system.

A lot of the advantages and features of the device server model are due to the adoption of OOP techniques. The main conclusion that can be drawn from this paper is that

1. the device access and control problem is adapted to being solved with OOP techniques,
2. OOP techniques offer a distinct advantage over traditional programming techniques for solving the device access problem.

8 Acknowledgements

The authors would like to express their thanks to the first generation of device server programmers (who programmed valiantly on even before *the doc* existed) – D.Carron, P.Mäkijärvi, T.Mettälä, C.Penel, G.Pepellin, M.Peru, B.Regad, B.Scaringella, M.Schofield, E.Taurel, R.Wilcke and H.Witsch. As the first generation of device server programmers they tested the device server model on a wide variety of devices and provided valuable feedback on the validity of the device server model. The device servers they wrote were used to build the ESRF machine control system and the first beamline control system.

References

[1] J.L. Laclare, "Overview of the European Synchrotron Light Source" in *IEEE Particle Accelerator Conference*, Washington, D.C., March 1987, pp. 417-421.

[2] "Off the MAP" in *Scientific American*, August 1991, pg. 100.

[3] P.J.Asente and R.R.Swick, *X Window System Toolkit*, Digital Press, 1990.

[4] K.E.Gorlen, S.M.Orolow and P.S.Plexico, *Data Abstraction and Object Oriented Programming in C++*, John Wiley & Sons, 1990.

[5] A.D.Birell and B.J.Nelson, "Implementing Remote Procedure Calls" in *ACM Transactions on Computer Systems* 2(1), February 1987.

Unix is a trademark of AT&T in the USA and other countries.

HP-UX is a trademark of Hewlett Packard, Corp.

SunOS is a trademark of SUN Microsystems, Inc.

OS9 is a trademark of Microware Systems Corp., USA.

Motif is a trademark of the Open Software Foundation, Inc.

NFS is a trademark of SUN Microsystems, Inc.

XDR is a trademark of SUN Microsystems, Inc.

The X Window System is a trademark of the M.I.T.

RTDB is a trademark of Automated Technology Associates, Indianapolis, USA

Oracle is a trademark of Oracle Corp.

Ethernet is a trademark of Xerox Corp.

All other trademarks or registered trademarks are of their respective companies. ESRF disclaims any responsability for specifying which marks are owned by which companies or organizations.

An Object-Oriented Implementation of the TRIUMF 92 MHz Booster Cavity Control System

N.A. Wilkinson and G.A. Ludgate
TRIUMF, 4004 Wesbrook Mall, Vancouver B.C., Canada, V6T 2A3

Abstract

A 92 MHz auxiliary accelerating cavity has been designed for installation inside the TRIUMF cyclotron, operating up to a maximum peak voltage of 200 kV. The cavity doubles the energy gain per turn for accelerating hydrogen ions in the energy region of 400-500 MeV, and reduces by 50% the stripping loss of the ion beam. The control system for the booster comprises a PC-based processor in a VME crate, for local control, and a 68030 processor with an ethernet connection as the interface to the TRIUMF Central Control System. The requirements for the booster control system were established by an object-oriented requirements analysis. Afterward, an object-oriented architectural design step was used to produce the processor allocation of the design, which was then implemented using C, for the VME processor, and a commercial database and screen generator product, for the VAX user interface.

I. INTRODUCTION

The RF Booster consists of an RF cavity, an RF amplifier, a transmission line connecting the amplifier to the cavity, a local control system and ancillary equipment. The RF cavity is made up of two symmetrical halves which are mounted on the lid and floor of the cyclotron vacuum tank and separated by 64 mm to provide a region free of all components. The accelerating voltage exerted by the RF cavity increases from 0 kV at a beam energy of 330 MeV to a maximum of 150 kV at a beam energy of 520 MeV. The RF cavity geometry is designed so that, when the booster is operating, each ion in the beam receives two impulses during its passage through the cavity. This energy gain enhances the turn separation of the beam, reduces the number of turns made during acceleration and reduces the beam loss due to electromagnetic stripping.

During the design and development of the booster, equipment specialists implemented a local control system based on an IBM-PC in VMEbus. Although this system met the requirements for local control of the booster by equipment specialists located in the RF area, its design did not consider the requirement for remote control by the cyclotron operators.

II. OBJECT-ORIENTED ANALYSIS

To identify the requirements for remote control of the booster, an object-oriented requirements analysis was carried out using the domain-driven specification techniques described in [1]. For this analysis, the notation of the Yourdon methodology with Ward-Mellor extensions[2] was used to represent objects, their behaviour and their interactions. Objects and their methods were represented by *data flow diagrams* (DFDs), object behaviour was represented by *state transition diagrams* (STDs) or *control transforms*, and information flows between objects were represented by *control* and *data* flows. A relevant aspect of the object-oriented methodology employed for the requirements analysis was the identification of objects which would interact with the RF Booster remote control system via information flows only. Such objects are known as *terminator* objects of the system.

As the only CASE tool available on-site was DECdesign, a CASE tool from Digital Equipment Corporation, this tool was used to capture the models of the booster remote control system. The strict Yourdon-Ward-Mellor methodology implemented by this tool did not easily accommodate the object-oriented paradigm chosen for the requirements analysis. As a consequence of this inflexibility, the desired goal of a purely object-oriented requirements document was contaminated by tool-specific considerations.

The DECdesign *verification* utility, however, proved invaluable in ensuring that control and data flows between objects were consistent and in ensuring that the data dictionary was maintained up to date.

III. THE RF BOOSTER REMOTE CONTROL SYSTEM

During the domain analysis it was recognised that personnel operating the booster would play two roles in their interaction with the booster: that of a *beam control* operator, whose only concern would be the affect of the booster on the beam, and the role of *equipment management operator*, whose concern was the management of the booster equipment.

These operator roles enabled the identification of the two domains spanned by the RF booster remote control system: the domain of beam control and the domain of equipment management. In the former domain, the requirements of the remote control of the RF booster by an *RF Booster Beam Control System* (RFB BCS) were examined and, in the latter domain, the requirements for remote management of the booster equipment by an *RF Booster Equipment Management System* (RFB EMS) were examined.

A. *The RFB Beam Control System*

Given the preceding assumptions, the purpose of the RFB BCS was described as follows:

- to enable operators in the Cyclotron Control Room to remotely control the phase and amplitude of the booster, and
- to provide operators in the Cyclotron Control Room with a meaningful display of the booster operating parameters necessary for remote control and monitoring of the booster.

The diagram of Figure 1 illustrates the interactions between the RFB BCS and its terminator objects. Since, by mandate, only the RFB EMS was allowed access to the booster via the IBM-PC in VMEbus, the RFB EMS was identified as a terminator object of the RFB BCS. Within the RFB BCS, the object *RFB Booster Control* was created to represent the interaction of the RFB BCS with the RFB EMS terminator object. The only other terminator object identified was the RFB BCS Operator, and the object *RFB Operator Control* was created to represent the interaction of the RFB BCS with the RFB BCS Operator.

The state transition diagram of Figure 2 illustrates the model of the RFB BCS operator behaviour elicited from personnel interviews.

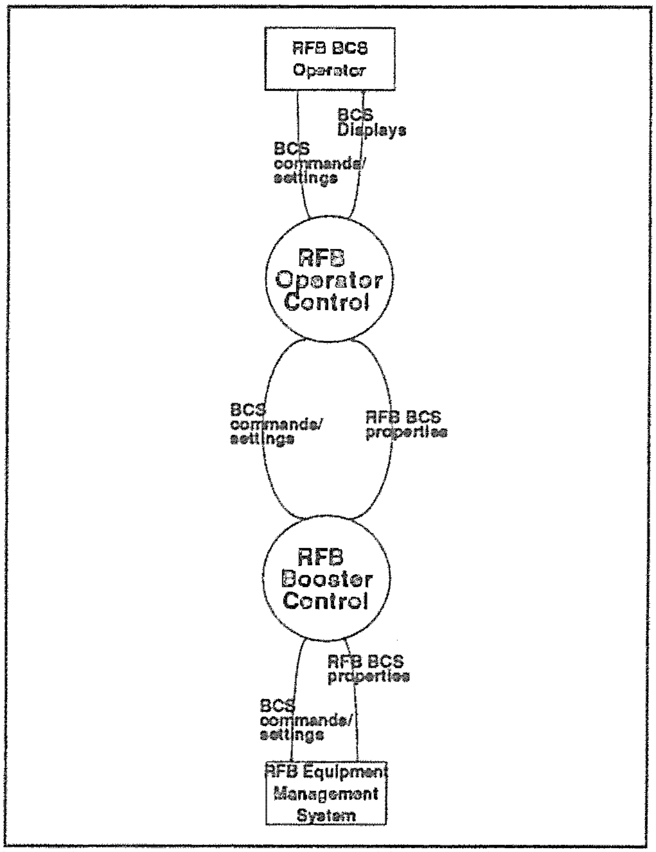


Figure 1: Beam Control System Objects (circles) and Terminator Objects (boxes)

The requirements analysis for the RFB BCS has presently completed two external reviews and is considered ready for design and implementation.

B. The RFB Equipment Management System

An RFB EMS operator would, at times, require privileged access to the booster equipment and, by mandate, such access was limited to the IBM-PC in VMEbus. It was recognised, however, that the RFB equipment management operator would require remote non-privileged access to the booster equipment in order to manage other aspects of the booster equipment, such as alarms, warnings and data logging. Thus, the purpose of the RF Booster Equipment Management System (RFB EMS) was established as:

- to enable operators in the Cyclotron Control Room to remotely manage the booster and its equipment.
- to provide operators in the Cyclotron Control Room with a meaningful display of the operating parameters required for remote management of the booster and its equipment.
- to provide a data logging facility for the booster, and
- to serve as the functional interface between the RF Booster Beam Control System and the booster equipment.

The RFB EMS was found to interact with four terminator objects: the RFB BCS, the RFB EMS data log, the RFB EMS operator and the RF Booster. For each of these terminator objects, a corresponding RFB EMS object was created to represent the interaction between the RFB EMS and the terminator object. These objects and the essential information flows between them are shown on Figure 3.

The RFB EMS requirements analysis has presently completed one external review. During this review, it became clear that, whenever excessive beam spill is detected by the RFB EMS, it must interact with the cyclotron ion source and shut off the beam to prevent damage to the booster and ancillary equipment. The RFB EMS requirements are presently being revised to reflect this new requirement.

IV. OBJECT-ORIENTED DESIGN

In the design phase, one of the major steps in implementing an object-oriented architecture is the allocation of the requirements specification to technology units. Several implementation choices which affect this allocation have already been made, namely:

- the operator interface for both the RFB BCS and the RFB EMS will be implemented via a VAX/VMS commercial real-time database and screen generator product obtained from Vista Control Systems.
- the majority of the RFB EMS and RFB BCS objects will be implemented in a VAX/VMS system, as nearly all of the terminator objects with which these objects must interact are accessible there.
- communication between the VAX/VMS system and the VMEbus system will be implemented via the *socket server* ethernet protocol suite described in [3]. This suite implements message-based inter-process communication between VAX/VMS processes and processes residing in a VMEbus processor running the Unison real-time operating system from Multiprocessor Toolsmiths,
- since all of the code running on the IBM PC in VMEbus runs under MS-DOS, an ethernet equipped VMEbus processor card running the Unison real-time operating system will be added to the VMEbus crate to provide network communication between the VAX/VMS system and the IBM PC in the booster, and

- the XDR (External Data Representation), which has been designated RFB1014 by the ARPA Network Information Centre, will be employed to define data formats for messages passed between the heterogeneous processors in the system.

In Figure 3, the aggregate information flow *RFB properties* consists of information which describes all properties of the booster. However, this information must be obtained from two different sources: the IBM PC in VMEbus and the TRIUMF Cyclotron Control System. The use of the message-based *socket server* network communication protocol allows information flows originating in the booster VMEbus system to be transparently encapsulated in messages between the VAX/VMS system and the booster VMEbus system.

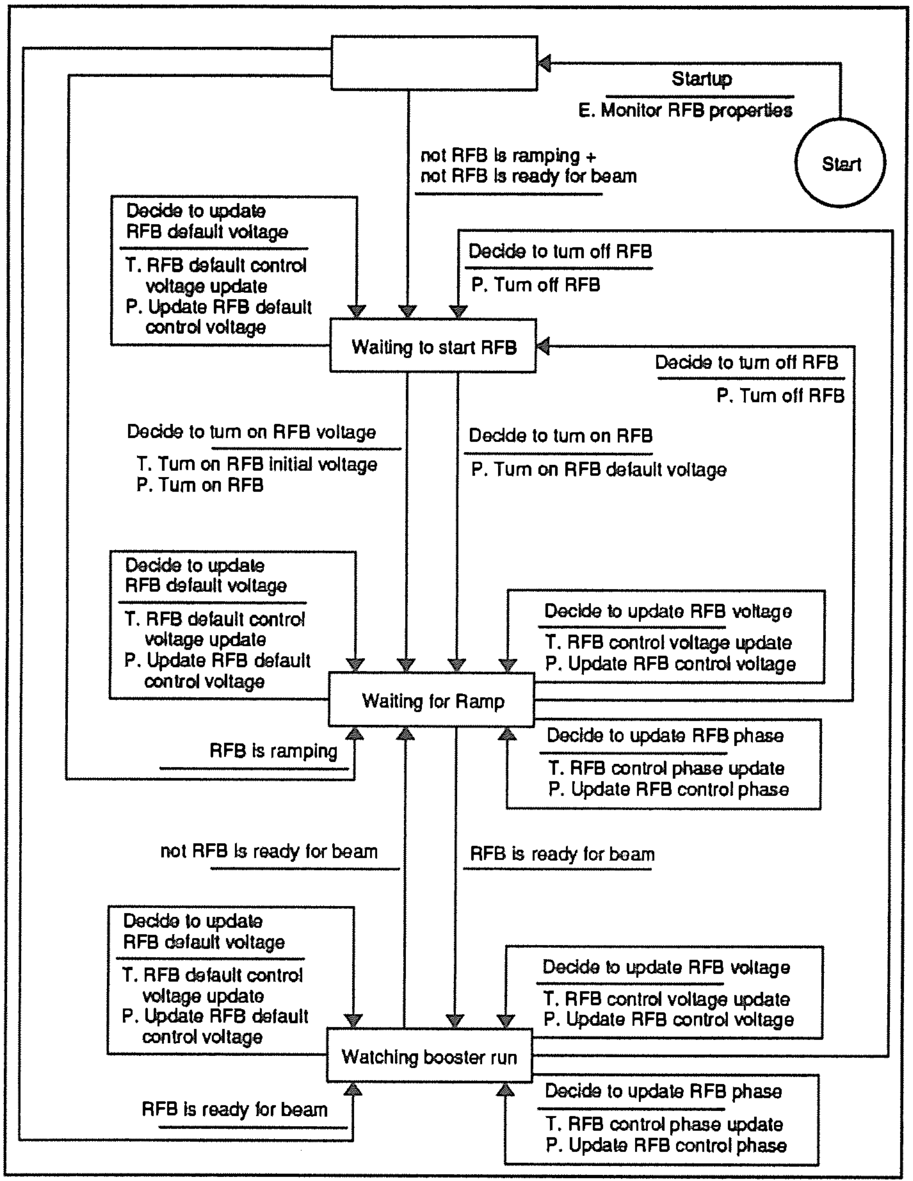


Figure 2: Beam Control System Operator Behaviour

V. OBJECT-ORIENTED IMPLEMENTATION

During the TRIUMF Beam Line 2C project, an approach was developed for the direct translation of object models into C code. The real-time data base from Vista Control Systems is central to the translation process - as inter-object data and control flows, including flows from terminator objects, are implemented via channels in the database.

When implemented by this method, each *object* is composed of the following code and data:

- a *private data structure* describing the object instance. Included in this data structure is a record of the object instance *state*, a list of the database channels accessed by the object and a *state transition matrix* which describes the object's state transition diagram in tabular form.
- a VMS Asynchronous System Trap (AST) handler, which serves as a *transaction centre* for object *events* and/or *conditions*, and
- the object *methods* which implement the object's behaviour as C function calls.

An object attaches its AST handler to each database channel from which it expects to receive a data or control flow, and the AST *user parameter* identifies the event which is associated with the channel. (In the case of condition or data flows, this event signifies a *change* in the condition or data flow.) An STD is directly translated: a transition condition on an STD becomes one or more events associated with database channels; the object instance state is maintained in private memory; and a transition action on an STD becomes the object method associated with a *state-event* pair in the state transition matrix. When an event occurs, the object's AST handler serves as a transaction centre by using the event and the object instance state to look up the *destination state* and associated method in the state transition matrix. The associated method is triggered by the AST handler and the destination state is recorded as a new value of object instance state. Once the function call to the associated method is complete, the AST handler checks for transition conditions which cause a transition to a new state. If a transition condition which causes a state transition is satisfied, the AST handler updates the object instance state and continues checking for conditions causing state transitions until none are found. When no valid transition conditions are found, the AST handler exits.

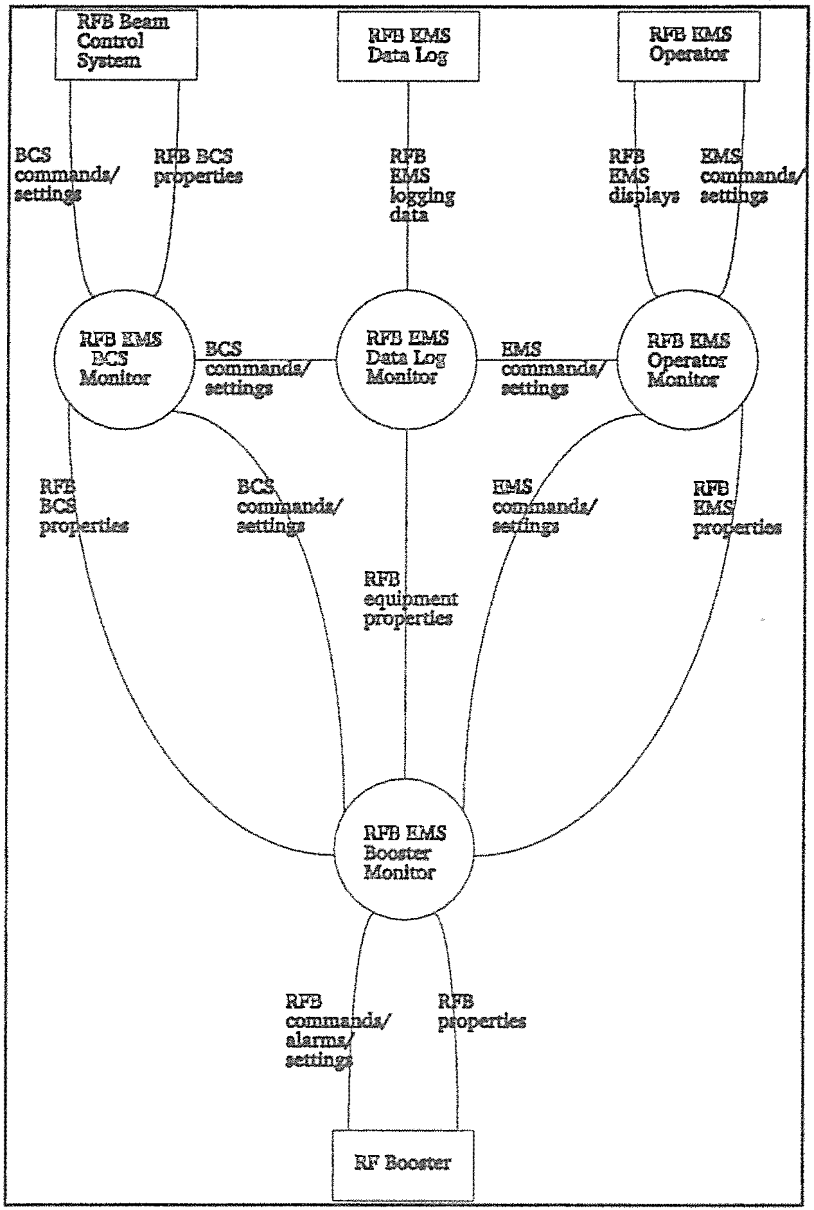


Figure 3: Equipment Management System Objects (circles) and Terminator Objects (boxes)

The considerable overheads inherent in VAX/VMS process creation precluded creation of separate processes for object methods which were continuously active. Instead, continuous activities were emulated by calling any enabled activities during execution of the AST handler.

The use of real-time database channels for information flows between objects enables concurrent and independent code development and testing. For instance, during the TRIUMF Beam Line 2C project, all information flows between objects were defined as database channels prior to the generation of any code. The coding and testing of objects did not depend on the existence of other objects because each object referred to appropriate database channels for its information flows, rather than to the objects from which the information flows originated. This enabled concurrent code development and testing on an object-by-object basis.

VI. CONCLUSIONS

The object-oriented analysis of the RF booster remote control system has produced a requirements specification which can be directly translated into code. However, the use of an inappropriate CASE tool (DECdesign) caused serious tool-specific distortions to enter into the specification for the booster remote control system.

The use of a message-based network protocol has enabled the transparent encapsulation of information flows which are obtained via the network. This has considerably simplified the design architecture of the RFB EMS by eliminating the need to distribute objects between processors.

The implementation of RFB EMS and BCS objects by direct translation of the requirements specification is presently being carried out. Object implementation is proceeding concurrently and independently, since channels in the real-time database used for information flows between objects can be defined before any objects are coded.

References

[1] C. Inwood, G.A. Ludgate, D.A. Dohan, E.A. Osberg and S. Koscielniak, "Domain-Driven Specification Techniques Simplify the Analysis of Requirements for the KAON Factory Central Control System," *Nuclear Instruments and Methods in Physics Research*, Vol. A293(1,2), Amsterdam: Elsevier Science Publishers, 1990, pp. 390-393.

[2] P. Ward and S. Mellor. *Structured Development for Real-Time Systems*, Yourdon Press, New York, 1985.

[3] N. Wilkinson and D. Dohan. "Synchronous Message-Based Communication for Distributed Heterogeneous Systems," submitted to: *International Conference on Accelerator and Large Experimental Physics Control Systems*, November 11-15, 1991, Tsukuba-shi, Ibaraki-ken, Japan.

The State Manager: A Tool to Control Large Data-Acquisition Systems

Ange Defendini, Robert Jones, Jean-Pol Matheys, Pierre Vande Vyvre, Alessandro Vascotto
CERN, Geneva, Switzerland

Abstract

The State Manager system (SM) is a set of tools, developed at CERN, for the control of large data-acquisition systems. A dedicated object-based language is used to describe the various components of the data-acquisition system. Each component is declared in terms of finite state machines and sequences of parametrized actions to be performed for operations such as the start and end of a run. The description, written by the user, is translated into Ada to produce a run-control program capable of controlling processes in a distributed environment. A Motif-based graphical interface to the control program displays the current state of all the components and can be used to control the overall data-acquisition system. The SM has been used by several experiments both at CERN and other organizations. We present here the architecture of the SM, some design choices, and the experience acquired from its use.

I. INTRODUCTION

Today's large data-acquisition systems are composed of an increasingly large set of programs which prove difficult to control. Furthermore, the different programs are not independent but co-operate and need to be synchronized: for example, they must be started and stopped in a given order. Finally, a system composed of many different programs is difficult to operate if one has to interact with each of these programs.

The SM [1,2] is a neat and flexible solution to this problem. It is a tool for building distributed run-control systems by means of a dedicated object-based language.

The system to be controlled is decomposed into a set of objects. Objects correspond to a part of the system: a program or a subsystem. Each object must then be described as a state machine, its main attribute being its current state. The state can take any value in a list of values declared by the user in his SM program. An object can interact with other objects by sending commands to them. The command triggers the execution of an action, which is terminated when the object reaches a new state.

Each activity of the data-acquisition system to be controlled should be handled by a single process. These processes are called associated processes because they are associated with an SM object. The SM communicates with them via messages handled by the OSP package [3]. The SM sends the commands triggering the execution of actions, and the associated processes reply when they assume a new state. These messages constitute the interface between the SM and the associated processes. The same interface is used by an overall control program to send commands to the SM itself as shown in Figure 1.

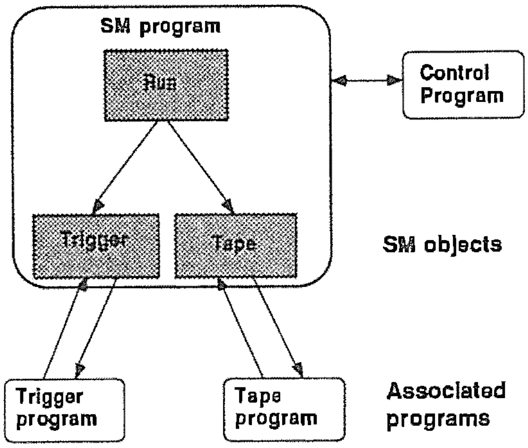


Figure 1. The SM program and the external world.

The communication package deals with distributed environments and thus allows commands to be sent to processes running on remote machines.

The objects are divided into two categories:

- The associated objects are associated with a program dealing with a device or an activity.
- The objects of the second category correspond to abstract entities that form part of the description of the system. They are internal to the SM.

The SM program written by the user is translated by the SM translator into Ada [4]. This Ada code is then compiled and linked to produce an executable image. The execution of this image will activate the run control and establish the communication with the associated processes.

II. THE SM LANGUAGE

A. *Object declarations*

The SM language contains declarations and instructions. The declarations are used to define the name of an object, its states, and actions. An example of a state machine for an object 'RUN' is given in Figure 2.

The corresponding SM declarations are:

```
object : RUN
    state : DORMANT
        action : START
    state : ACTIVE
        action : PAUSE
        action : STOP
    state : PAUSED
        action : RESUME
```

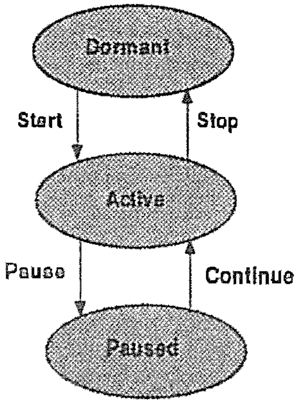


Figure 2. Example of a state machine.

The execution of an action is different, depending on whether the object is associated or not. For an associated object, the execution of an action consists of sending a command to its associated process. When the process has terminated the execution of this action, it will return to its new state, which is mimicked by the associated object. For a normal object (not associated), the execution of the action consists of the execution of a sequence of SM instructions.

B. The instructions

Four basic instructions are contained in the language.

The DO instruction is used to send a command asynchronously to an object. The sender carries on with its own execution after the command has been sent. The command is put into a queue if the receiver is not ready to execute. A queue of pending commands is maintained for each object in the system. The next command is delivered to an object when this object is ready to accept it, i.e. when the object is in a stable state and is not executing any action.

The sequence of instructions corresponding to an action is terminated by the instruction 'TERMINATE_ACTION /STATE=state name'. This instruction can be placed anywhere in the code, thus stopping the execution of the code and putting the object in a stable state specified in the instruction.

The IF instruction tests the state of one or many objects and combines the results in a logical expression. All the commands present in the object queue must be executed before testing the object state. The IF instruction synchronizes the object executing the IF statement with the objects whose state is tested in the instruction. The language also specifies a special state, the 'dead state', which is assumed by an associated object when the program associated to it is not running. This special state allows testing in the SM code whether the associated program is running or not.

The WHEN instruction triggers the execution of an action spontaneously when a logical expression based on the states of objects becomes true. This instruction is used to react asynchronously to a state change in the system.

The example below uses the four basic SM instructions:

```
object : RUN
    state : DORMANT
        action : START
            do MOUNT TAPE
            do START TAPE
            do ENABLE TRIGGER
            if (TAPE in_state WRITING) and
                (TRIGGER in_state ENABLED) then
                terminate_action/state=ACTIVE
            else
                terminate_action/state=FAILURE
            endif
    state : ACTIVE
    when TAPE in_state END_OF_TAPE do STOP
        action : STOP
            do DISABLE TRIGGER
            do STOP TAPE
            do DISMOUNT TAPE
            terminate_action /state=DORMANT
    state : FAILURE
        action : RESET
            do RESET TAPE
            do RESET TRIGGER
            terminate_action /state=DORMANT
            ...
```

C. The SM domain and the visible objects

The object name has to be unique in one SM program because the object must be addressable unambiguously. However, this may be a limitation in big systems composed of the repetition of similar subsystems. It may also be easier to divide a big system into smaller SM programs. This is what the SM domains are for (Fig. 3). The SM domain is a logical domain that consists of one, and only one, SM program and its associated programs. The SM domain limits the visibility of an object. The name of the object must be unique in one domain, but the same object name can be used in different domains.

An object belongs to one domain only, but it may be rendered 'visible' to outside domains. One SM program can therefore be controlled from another SM program. Figure 3 shows an example of a top-level SM controlling two other SMs in different domains.

The way to invoke an object of another domain is to specify explicitly its domain as shown in the example of Figure 3: the SM program of the domain MAIN contains references to the objects 'TPC::RUN' and 'HCAL::RUN'.

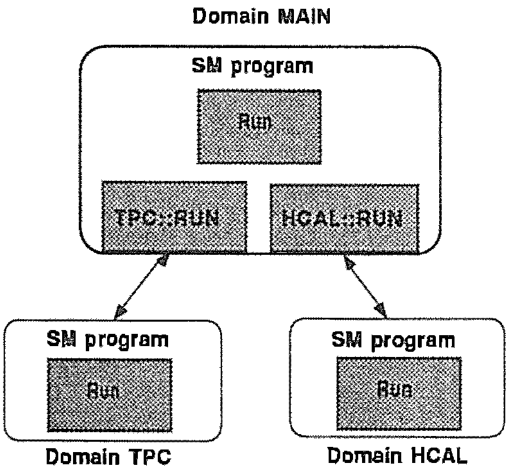


Figure 3. Example of use of the SM domain.

D. *The classes and the access objects*

With the declaration statements seen up to now, the user has to declare each object in the SM program. However, it is quite common to have systems where many objects are identical, but not their names. The notion of class has been introduced to cope with this case. A class of objects is similar to a data type in standard programming languages. The declaration of a class must define the class name, its states, and its actions. Once a class has been defined, it is sufficient to invoke the class name in the declaration of the objects that belong to that class. The following example shows the declaration of a class 'TAPE', and of two objects 'TAPE1' and 'TAPE2' of the class 'TAPE':

```
class : TAPE /associated
    state : AVAILABLE
        action : MOUNT
    state : MOUNTED
        action : DISMOUNT
object : TAPE1 is of class TAPE
object : TAPE2 is of class TAPE
```

This feature improves the readability and the 'maintainability' of the code, and reduces the size of the declarative part of the program.

A special type of object has been introduced to handle an object belonging to a class: the access object, which is like a pointer to any object of a given class. The 'access' statement specifies which object of the class is being accessed. The basic SM instructions can use the access object to refer to one object of a class indirectly:

```
object : CURRENT_TAPE is access to class TAPE
        state : NOT_USED
            action : SELECT_TAPE1
                access TAPE1
                ...
    do MOUNT CURRENT_TAPE^
    if (CURRENT_TAPE^ in_state MOUNTED) then
    ...
```

E. *Command parameters*

Some associated programs may need parameterized commands. Parameters are specified in the SM code as simple strings or as logical names translated at run-time. The parameters are appended to the string of the command before it is sent to the associated process. An example is given below.

```
do MOUNT ("/LABEL=" VOLUME) CURRENT_TAPE^
```

III. THE STATE MANAGER AND THE EXTERNAL WORLD

A. *The control program*

The SM program itself can be controlled at run-time by a 'control program', which can send a command to the SM with a call to a subroutine. The control program can also examine the current state of the system. A library of routines is available for the communication between the control program and the SM.

A general purpose control program has been built using the Motif graphical user-interface toolkit. Each object is shown on the screen as an icon. The user interacts with an object by clicking on its icon to reveal a popup menu. The user can send a command to the object, see the object's past states and actions, and view the queue of actions to be performed by an object. The display shows one domain at a time. This domain can be selected by the user.

Many aspects of the display can be customized by the user; objects icons can be hidden, moved and replaced by user-defined ones. The user's personal configuration can be saved and restored automatically when the display program is restarted.

B. *The associated programs*

The associated programs running under the control of the SM must conform to a well-defined interface. They must be command-driven and send back their state when it has been modified. A library of routines is available for the communication between the associated processes and the SM.

The simplest structure of an associated program is as follows:

```
C   Initialization call
    call SMI_INIT
C   Associate the program with an object
    call SMI_ASSOCIATE (object_name)
    do while (program active)
C       Receive next command to execute
        call SMI_GET_COMMANDW (command)
C       Decode the command
C       Return new state after execution
        call SMI_TERMINATE_COMMAND (state)
    end do
```

C. *The multiple-state associated objects*

Some associated programs may be difficult to describe in terms of a state machine with a unique current state. It is possible to divide them into sub-objects, each of which have their own state. A sub-object can be a device that the program has to deal with, or a level of alarm, and so on. The result of this division into sub-objects is that the object itself has many concurrent states

Extra SM instructions, not described above, are required to fully define object's state machines. A tool exists to help the user generate associated programs state machines.

IV. IMPLEMENTATION

The translation of the SM language into Ada proceeds directly by the semantics of its main instructions.

Each SM object executes its actions in parallel with the other objects. Each object is translated into an Ada task which is an independent thread of code. The current state of an object becomes a variable with a value equal to one of the value of an enumerated type. Actions are translated into Ada rendez-vous, where two tasks are synchronized.

However, the SM action is asynchronous whereas the Ada rendez-vous is synchronous. Therefore, for each object, a second Ada task is introduced to handle the queue of pending actions, to disable its access and to produce the asynchronism needed for the SM action. The two Ada tasks corresponding to an object are shown in Figure 4.

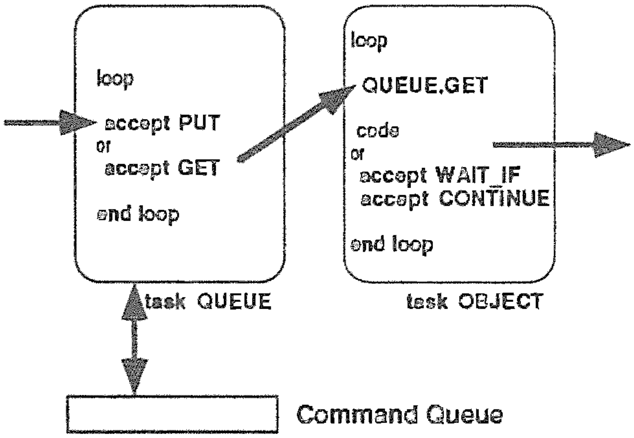


Figure 4. The Ada code corresponding to an object.

As described before, the IF instruction requires some synchronization between the object executing the IF statement and the objects whose states are evaluated in the condition. Two rendez-vous of the task 'Object' block and unblock the execution of an object at the beginning and the end of the IF statement.

For example consider the following SM instructions.

```
object : RUN
...
        do MOUNT TAPE
        if (TAPE in_state MOUNTED) then
```

The DO instruction is translated into a rendez-vous with the QUEUE task to add an action to the queue of the destination object. The IF instruction causes the TAPE object to block, its state to be evaluated by Run, and then to unblock.

```
QUEUE(TAPE).PUT(MOUNT)
QUEUE(TAPE).PUT(WAIT_IF)
if (OBJECT(TAPE).STATE=MOUNTED) then
endif;
OBJECT(TAPE).CONTINUE
```

In addition, a dedicated Ada task consists of all the WHEN instructions contained in the program. This task is scheduled each time an object assumes a new state. The task evaluates all the WHEN conditions and adds the appropriate actions to the queues.

V. CONCLUSION

The SM is a new approach to the problem of run control. It has proved to be both flexible and reliable, during its use at CERN in collaborations such as DELPHI, OBELIX, and Omega.

By coding associated programs according to simple principles, SM provides an object-based approach to DAQ design that benefits the control and maintenance of the system.

VI. REFERENCES

[1] J. Barlow et al., "Run Control In Model: The State Manager", in *Proc. 6th Conf. on Real-Time Computer Applications in Nuclear, Particle, and Plasma Physics*, Williamsburg, Virginia, 1989 [IEEE Trans. Nucl. Sci. NS-36 (1989) p. 1549]

[2] A. Deffendini, B. Franek, P. Vande Vyvre and A. Vascotto, The state manager user manual, CERN-ECP internal note, Geneva. This document can be obtained, upon request, from the authors, c/o CERN.

[3] R. Jones, OSP user's guide, CERN-ECP internal note, Geneva. This document can be obtained, upon request, from the author, c/o CERN.

[4] Ada Joint Program Office, United States Department of Defence, *Reference Manual for the Ada Programming Language*, ANSI/MIL-STD-1815A, Washinton D.C.: Government Printing Office, 1983.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S15CSE01

CASE in CERN's Accelerator Sector

A. Albrecht (SL), A. Cabas-Alonso (AT), F. Chevrier (ECP), A. Daneels (AT), Ch. Delamare (CN),
G. Ferran (CN), S. Foffano (MT), P. Heymans (AT), J.P. Matheys (ECP), D. Manglunki (PS),
Y. Marti (SL), D. Missiaen (AT), G. Moorhead (CN), O. Novakov (AT), T. Pettersson (PS),
J. Poole (SL), M. Pozzato (SL), J.P. Quesnel (AT), S. Santiago (CN), J. Schinzel (AT),
N. Segura-Chinchilla (AT), C-H. Sicard (PS)
Presented by A. Daneels (AT)
CERN, European Organization for Nuclear Research, 1211 Geneva 23, Switzerland

Abstract

As in the software industry where computer aided software engineering (CASE) methodologies and tools are commonly used, CERN endeavours to introduce this technology to improve the efficiency of designing, producing and maintaining software. A large project is currently under development in the administrative area whereas a dedicated group has been set up to evaluate state of the art techniques for software development relating to physics experiments. A similar activity, though on a smaller scale, has been initiated in the accelerator sector also in view of the large amount of software that will be required by the LEP200 and the LHC projects. This paper briefly describes this technology and gives an account of current experience with the use of CASE methods and tools for technical projects in the accelerator sector at CERN.

1. INTRODUCTION

Software engineering is the application of techniques which lead to the implementation of better quality software. It implies a planned process of producing well-structured, reliable, good quality, maintainable software systems which corresponds to the users' needs, within reasonable time frames [1].

This definition suggests that software engineering includes a good deal more than just producing computer programs and that good software development includes documentation, databases, operational procedures, etc. Furthermore, it focusses the planned aspect of the process: as any other engineering discipline, software production should be properly managed with scope definition, specification analysis, cost estimation, production plans, role distribution, etc.

According to industry statistics, 75% of custom software development projects are rejected because they came either too late to be useful or did not correspond to the users' needs. However the complexity of application software grows continuously and today's average business package takes 32,000 man-days, i.e. 160 man-years to develop [2]. This is not dissimilar to the effort spent at CERN on application software for accelerators. Indeed, in the eighties at least 500 man-years were invested on controls and database applications for the PS accelerator complex, SPS and LEP, not including numerous developments that have not been accounted for. The annual maintenance effort is estimated to be around 15% of the development effort and exceeds the production capacity of the groups in charge. Here, maintenance is defined as software repair and update resulting from a changed functional specification of the software product. For each new development the volume of the software increases because more sophistication is required. By the time the LHC is approved, the demand for application software may well be two to three times higher because of increased functionality, increased information volume, more severe execution time constraints due to the superconducting nature of the machine, and higher reliability [3]. Even if the groups in charge manage to develop such large packages with the help of professional, voluntary and temporary staff, they will only be able to maintain it if software of sufficiently good quality is produced so as to dramatically reduce its maintenance cost.

2. WHAT IS CASE ?

CASE stands for Computer Assisted Software Engineering.

For each new software project the engineer is recurrently performing a number of similar activities: collecting information from his client, organizing that information, cross checking with the client, etc. The process is systematic, iterative and proceeds in increasing degree of detail. A number of methods and procedures could be derived which, because of their recurrence, would be more efficiently executed if assisted by computer programs. Software tools were thus developed to assist the software engineer in collecting, organizing, storing, retrieving and cross checking that information throughout the development process of his project. The information is introduced through graphical and alphanumerical user interfaces and recorded in a repository, possibly a database management system, so that it is available throughout the production life cycle for complementing, checking and various administrative operations.

CASE is introduced in order to encourage better quality designs, to increase productivity and to render software projects more manageable. Better quality software leads to reduced maintenance: industrial companies e.g. BBC, now ABB, claim to be able to reduce the maintenance to 2% of the development cost by using such tools [4].

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S15CSE01

3. CASE PRODUCTS ON THE MARKET

There exist a large variety of CASE products on today's market: Computerworld [5] published a list of not less than 33 such products that were selected on the basis of the tools they provide for analysis, design, code generation, debugging, code / configuration management, testing and integration, code analysis, maintenance, documentation, reverse engineering, project management, etc. Unfortunately, no tools exist yet that provide all these facilities in a single environment, and as they are developed by different vendors, they are often not compatible: e.g. the output of a first vendor's analysis and design tool may not be usable by another vendor's code generator. Therefore some of the larger vendors endeavour to design integrating frameworks through which tools from multiple vendors can communicate. Such frameworks are called Integrated Project Support Environments, IPSE. However, even IPSE are only effective provided the IPSE producer has established contracts with third party CASE developers.

To date the application software market seems to be subdivided in two major domains: database and real time applications. Whereas database applications focus the structuring of information so that it can be easily retrieved and processed in a variety of associations or combinations as are useful to a business, the real time applications, in particular the control applications, concentrate on the functional and temporal aspects of the process: what functions have to be performed - and when - to respond to the process' behaviour. This difference is reflected in the facilities that are provided by current CASE tools: those for database applications provide extensive facilities to model the data, whereas the tools for real time applications are more concerned with modelling the functional and temporal behaviour of the process. One would expect tools for both database and real time applications in future to converge towards a single environment.

4. CASE METHODS

CASE methods cover two aspects: a technical one that is concerned with the analysis, design and implementation of the application, and a managerial one that provides various project control and quality assessment techniques with rules for a proper organization of the project team where every member has a well defined role, in addition to recipes for the organization of reviews, feedback sessions with the users, quality assurance sessions, etc.

As an initial step, most CASE methods start with defining the objectives and the boundaries of the project. Next they enter into a strategy phase where a model is produced of what needs to be developed and describing its relation to the environment. This model is then worked out in more detail during the analysis. Until this stage there is no consideration of how the requirements will be fulfilled, this is the purpose of the design. Next comes a build phase followed by implementation and commissioning. It should be noted that these methods bear strong resemblance with those developed in other engineering disciplines.

5. CASE TOOLS

CASE tools help the developer in reaching a full understanding of his project. They guide him through the various phases of the development process so as to produce a series of models to entirely depict his program: they are the "blue prints" of the program he will implement.

Software involves in general three basic components: data, processes that operate on data, and the time or events at which the processes are executed. CASE tools allow to model respectively the structure and the internal relations of the data, the functions and the events; they further depict how data relate to functions, functions to events, etc.

The models are represented by diagrams constructed by means of a number of generally accepted graphical notations: e.g. Yourdon / DeMarco for control flow and state transition, Jackson for data structure, Chen, ERD (entity relationship diagram) or NIAM (Natural Information Analysis Method) for entity relationships, Stevens, Myers and Constantine for structure charts. CASE diagrams thus appear to the reader as roadmaps through which he can navigate to understand the structure of the program and the functionality it provides. These diagrams are complemented by structured textual description of e.g. entities, attributes, processes, tasks, events, etc. and free format specification. Textual specification are also entered through appropriate editors. The more advanced tools further provide extensive checking such as referencibility of entities, completeness and consistency of the diagrams, identifying e.g. unlabeled or unbalanced flows, entities and attributes , etc.

Most CASE tools support multi-user development and run in a distributed environment. Their user interfaces are based on graphical and textual editors with Macintosh or MOTIF style interaction, through which the programmer enter its specifications that are placed in a central repository. The repository integrates the various tools into a single environment (hence i-CASE, i.e. integrated CASE); data are available at each phase of the project life cycle and can be checked against the functionality for completeness and consistency. Project administration tools are also provided to define access rights to the repository for each member of the project team depending on his privileges, to control the versions of diagrams, etc..

6. PILOT PROJECTS AND CASE TOOLS USED IN THE ACCELERATOR SECTOR

In order to evaluate the applicability of CASE in the accelerator sector at CERN, a number of pilot projects have been selected in various fields relating to accelerators: controls, data acquisition, cryogenics, modelling, radiation monitoring and survey.

The pilot projects for database applications concern new designs, retrofitting existing systems or modelling for documentation purposes. ORACLE*CASE was selected because it provides a full information system engineering environment based on the ORACLE relational database that is standard at CERN. However, in order to compare the ORACLE methodology to model entities with NIAM, a "shadow" exercise has been initiated using RIDL, Relational IDea Laboratory, of Intellibase NV (Antwerp, Belgium), that is based on the latter methodology.

The evaluation of CASE for real time applications has focused on StP, Software through Pictures produced by Interactive Development Environments, IDE (San Francisco, USA) and Real Time Engineering Environment, RTEE, of Westmount Technologies (Delft, the Netherlands). These CASE tools complement Teamwork of CADRE Technologies that is in use in SL division[1].

Brief Description of Pilot Projects

- **Data base Applications**

The database for cryogenics should provide a full inventory of the equipment and information on the state. It should also provide information concerning control signals and algorithms and incorporate archive measurement and test data of the various instruments for maintenance purposes.

The Survey group is responsible for the proper alignment of approximately 10,000 accelerator and transfer line elements over approximately 60 km. Most of that information is stored in a central ORACLE database and should be extended to include data about the stability of the measurement devices.

The SL database aims at providing a central description of the SPS and LEP accelerators, reference information on the state of the machine and at maintaining historical reference data for a variety of applications in different areas: controls, vacuum, survey, beam instrumentation, magnets, power converters, radio frequency, mechanical design, and accelerator physics.

- **Real time applications.**

The LEAR control system is currently running in a VAX/VMS environment with UIS as console interface software. The front end hardware is based on PDP with Pascal software. By the accelerator start-up in 1992, the front-end part of this control system should be adapted to an X-system environment and its database migrated into an ORACLE one. It was intended first to model the control system by the use of ORACLE*CASE for documentation purposes, and next, to design an enhanced version.

The General Supervisory System monitors the environmental conditions in the four experimental sites of LEP through more than 1,000 detectors. The system is based on an expert system with its knowledge base and security rules stored in an ORACLE database. It grew as experience accumulated over the years and has now reached a point were a major overhaul is needed to rationalise and enhance its functionality. The need to port the system to a UNIX environment provides the opportunity to undertake its retrofit.

For StP two parallel evaluations were carried out. The first one concerned an asynchronous data-collector service that will be incorporated within the control system of the PS accelerator complex. That system has been analysed with the aid of data structure, data flow and state transitions editors. The second evaluation concerned the use of structure chart editing facilities for the detailed design of an error logging program.

In contrast to this, the evaluation of the RTEE for real time application was not based on a real life project. Instead, the idea was to analyse the facilities the tools provide, in some cases by modelling parts of existing real time programs, and by evaluating the method and notation on which they are based.

Objectives

The prime objectives of the pilot projects are to gain experience with CASE methodology and tools and to evaluate their applicability to technical projects. All projects, except the one using RTEE, aim at producing real applications. This was in particular a pre-requisite for being entitled to an ORACLE*CASE licence.

From a management's point of view, CASE is evaluated as a way to produce better quality software and to enhance communication with the user, cooperation across projects and progress visibility. It is also aimed at economy of scale by sharing the resources invested in the evaluation exercises.

Constraints

The mixed nature of the projects reflects the constraints.

The pilot projects have limited financial and human resources: even if the overall team involves around 25 persons, depending on their role some are only spending a fraction of their time in the project. In addition the teams belong to different Divisions: Accelerator Technology, Computers and Networks, Electronics and Computing for Physics, Mechanical Technology, Proton Synchrotron, SPS-LEP. They are thus geographically distributed and their members are also involved in other activities whose priorities depend on the local divisional objectives. The teams cover a wide range of disciplines, each having it own habits and jargon. However, this multidisciplinary, multidivisional and part time nature are typical for the way projects are often carried out at CERN.

---

[1] Note: Teamwork, that was recommended in 1989 by CERN's Technical Board for Process Controls and Accelerator Electronics (TEBOCO), became unusable for PS division when it was decided to standardize all CERN controls on UNIX or Ultrix. PS division was equipped with VAX stations and their VMS O/S was replaced by Ultrix: although operational on VAX-VMS or DEC-Ultrix platforms, Teamwork appeared not to run properly on PS' hybrid VAX-Ultrix configurations.

In addition the projects are different in size: though most are small and well encapsulated, the database project for cryogenics appears as a large one. It was thus difficult, not to say impossible, to keep the pilot projects synchronised so as to reach a more homogeneous level of expertise, possibly sharing some common designs, agreeing on a common glossary, etc.

The project involves a mix of platforms: VAX stations running VMS, DEC and SUN-SLC & IPC running Unix. The workstations involved in the evaluation of ORACLE*CASE host the tools and are linked by ethernet to a central VAX 6420 with VMS 5.4 that houses the central database and the dictionary. The workstations communicate with the central engineering database through SQL*NET (Fig.1.).

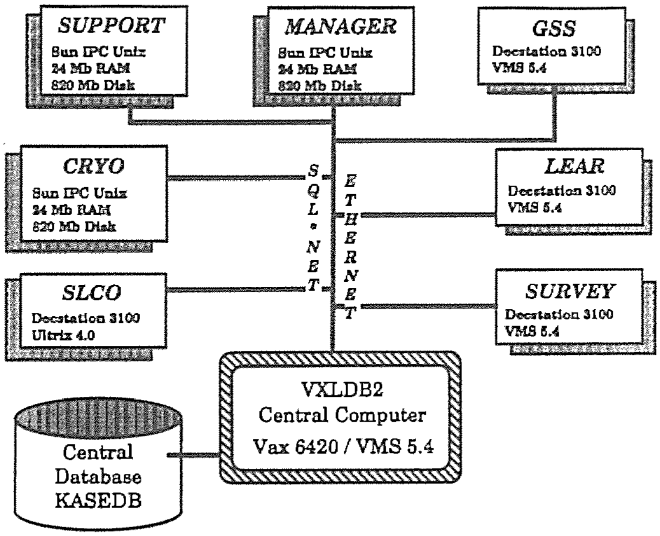


Figure 1

On ORACLE Corporation's request a tighter management scheme was adopted for the database projects than for the real time ones.

Evolution of the projects

It was felt that learning the method and the tools by reading manuals and playing with the tools alone would be too time consuming. Therefore systematic training on the method and the tools was introduced for the database pilot project teams and also for those involved in real time CASE evaluation who wished so. Professional consultancy was called upon to define the scope of the projects, for formal reviews, feedback and quality assurance in order to rapidly reach a reasonable degree of expertise within the CERN software engineering team.

In general the methods were followed but needed to be adapted to the way of working, proper to scientific organizations. This is particularly true for ORACLE*CASE Method whose rigor was difficult to accept since it broke with the usual individual style of working. The methods for real time applications adapt easily to a more familiar bottom up design without excluding the rigid top down approach.

The Survey and SL database projects are currently in the build stage and are generating tables and forms. Those involved in retrofitting the database for survey claim that by using CASE they were able to optimize their design so as to improve the performances of data retrieval by a factor of 5 when compared to the original design.

The cryogenics database however grew to become a much larger project than anticipated. This is in part due to the fact that an overall picture was originally not available and only became apparent as one proceeded through the method. Much time was spend on understanding the cryogenics problems in addition to learning the method and the tools. However, at this time a model has been designed that is reasonably stable and has obtained agreement with the users.

LEAR failed to model its control system by using ORACLE*CASE. Despite hopes to model the system for documentation purposes, it turned out impossible to describe the time dependencies and the process sequencing that are fundamental aspects of real time systems.

Eventually modelling the General Supervisory System could not be pursued because of lack of manpower. However, it came to a point where similar problems as for LEAR became apparent because of its strong real time nature.

7. COMPARING METHODS AND TOOLS

*ORACLE*CASE*

ORACLE*CASE Method follows a top down approach whereby a number of tasks have to be performed in a specific sequence. These tasks are grouped in the various project phases which must yield well defined deliverables before one can proceed to the next phase. The method also includes many cross checking techniques to ensure the accuracy, consistency and completeness of the design. It also provides techniques and procedures for team and project management emphasising the user involvement throughout the life cycle and control techniques.

The following phases are identified:

Scoping to define the limits of the project, its objectives and constraints. ORACLE puts a lot of emphasis on this phase as a means to keep the project on its tracks.

Strategy: in this phase a model is produced of what needs to be developed. The functionality is represented by a hierarchical breakdown of the functions to be performed and a structural model is built with the entities. This phase also includes statements about quality standards to be achieved. The requirements are next translated into written specifications, drawings, data sheets, etc.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S15CSE01

Analysis: the previous model is now worked out by adding attributes to the entities and by describing them in detail. The data flow diagram describes movement of data between the functions. Matrices allow the cross checking of completeness of the relations between functions and entities, entities and attributes.

Design: this phase concentrates on **how** the detailed requirements, as defined in the strategy, will be fulfilled. An information system architecture is produced identifying the various applications covered by the functions that will access the database together with a detailed program specification.

Build stage: the system is now built and reviewed with the user.

Transition phase: the implementer provides the customer with the necessary support to ensure a smooth transition from the old system to the new one.

Production (Operation): ensures the running of the system whilst its performances are being monitored.

It should also be stressed that ORACLE*CASE puts a lot of emphasis on the management aspect of the project and on the relation between the information system engineer and the customers.

The tools provided by ORACLE are CASE*Dictionary, an internal multi-user database that acts as the central repository for all information relating to a project through all stages of its life cycle. This dictionary is filled through diagrams (function hierarchy, entity relationship, data flow,...), matrices and interactive forms provided by CASE*Designer. Furthermore, CASE*Generator generates tables, menus, interactive forms using the 4 / 5 GL ORACLE tools, and reports. The central part in ORACLE*CASE is the entity relationship diagram, that is based on an extension of the Chen method. It provides a large number of facilities and coded symbols to describe the entities, their type and attributes as well as their relations.

CASE for Real Time Applications

The rigid top down approach that is followed by some CASE methods and tools has been the object of endless debate and has been abandonned by some of its major initiators in favour of a more pragmatic middle out approach (6) or an object oriented one. The tools that were evaluated within the scope of this exercise, StP and RTEE, follow this middle out approach and allow the designer to progress both towards higher levels of abstraction and lower levels of details in the course of his analysis and design. Except for this particular feature, the CASE method for real time applications follows essentially the same sequence of phases (see previous paragraph) by which a series of models are produced that are gradually refined and complemented so as to ultimately yield template code. However, the user involvement and team organization during the project life cycle is not defined to the same extent as by the ORACLE*CASE method.

The major steps through the method are:

Survey, evaluating the requests and the feasibility within budget and time constraints.

Analysis, defining the **environmental model** of the system (i.e. how does the system relate to its environment) and the **behavioural model** showing the processes inside the system.

Architecture, where **processes**, **tasks** and eventually **modules** are modelled. This is a not so common feature that is particularly useful in case of distributed multitasking systems.

The **strategy phase** of ORACLE*CASE appears to include the **survey** phase with some overlap in the **analysis** when defining the environmental model.

Contrary to ORACLE*CASE whose diagrams are limited to describing the functional hierarchy, relationships between entities and dataflows, the **tools** for real time application development provide diagrams to model several control aspects of the system.

The **control flow diagram** provides hierarchical models of the system's functionality showing data and control flow between processes, the processing of data and their control actions. It is based on a number of rules and symbols to represent data processes and control processes, types of flows (data flow, update flows, continuous data flow, control flow and continuous control flows), stores (data and control stores such as on/off information), and external processes with which the system communicates. The familiar **data flow diagram** one finds in ORACLE*CASE, appears as a specific view in the control flow diagram. It is complemented by a **list of** events (temporal, control or flow oriented) identifying the stimuli that occur in the external world and to which the system has to respond, and diagrams to illustrate how these events relate to the system.

The **data structure diagram** gives an abstract and static representation of the data: it shows how the data are structured hierarchically, not their relation. It allows specification of simple sequences, iterations and selections.

The **entity relationship diagram** represents the static relations between entities using the Chen modelling technique.

The **state transition diagram** describes the states of a system and the sequence of activities between the states (in our case, following the Yourdon method).

The **system architecture diagram** is an extension of the control flow diagram. It allows one to graphically assign and partition tasks to processors and incorporates symbols for processors, tasks, interrupt service routines, message queues, message boxes, event queues and event flags.

The **structure chart** depicts the functional breakdown of the system. A module is defined as a collection of program statements.

Ultimately the code is generated in three steps: generation of a **program design language, PDL,** from the structure chart; generation of **program code** from **PDL**; generation of data type declarations from data type diagrams.

8. EVALUATION CRITERIA

CASE products are evaluated relative to their architecture, the environment in which they run, the applications that are provided with the product, the tools themselves and, last but not least, a number of economical considerations relating to the vendor.

The architecture is expected to be "open" allowing the user to customize e.g. menus, commands, etc., to set defaults and to extend the tool by invoking other tools such as e.g. testers and simulators.

The environment in which the tools run should be based on standard operating systems, windowing system; they should also support heterogeneous networks.

The product should allow tracing the requirements throughout the various phases and provide facilities to produce standard documents that can be customized, etc.

The tools themselves are evaluated on the ergonomics of their editors and the techniques they support.

Selecting such a tool implies a strategic choice and one has to take into account vendors' "health" on the CASE market, the support he is able to provide, and the evolution of his product with regard to those of competitors.

9. PRELIMINARY CONCLUSIONS

Although it is premature at this stage of the project to draw definite conclusions, sufficient experience has been accumulated to be convinced of the usefulness of CASE methods. They are very rigorous and convey confidence in the quality of the product. The clients admitted, after initial skepticism, that the method provides a sound basis for discussion. Even if during the analysis some projects grew larger than anticipated and extended well beyond the part to be implemented, it shows that an overall picture has been obtained thanks to the method and that provision for coherent future extension will be built into the model. In addition the methods give good control on the project life cycle.

Despite a number of weaknesses and inconveniences that the vendors endeavour to correct, all products proved to be state of the art. They provide an excellent basis for communication: designers and users discuss over a full set of specifications both graphical and textual, with agreed definitions and terminology avoiding misunderstanding. CASE appears thus as a must for large projects, and prepares the ground for subcontracting. Though at present CASE may seem an overkill for smaller projects, it may be that as soon as the technology is well mastered it allows to produce this type of project in "no time".

No major problems were encountered when working in a mixed environment. Most tools run on workstations and access the central repository installed on a server through the network.

None of the tools for real time applications that were evaluated run with ORACLE as internal repository. This is a limitation in CERN's context where ORACLE is a standard, but no tool could as yet be found on the market that satisfies this requirement.

However, the use of CASE implies a real change of habit: it requires an analytic approach and a disciplined style of working that contrasts with the previous free style. Also it needs a relatively long learning curve: the longest probably for ORACLE*CASE that often requires a good insight of the method and its relation with the tools for efficient understanding; therefore training by the vendor is of great help. Nevertheless, even after training, one should be prepared to invest significant time in order to become fluent.

It was difficult to come to an agreement on the method amongst the participants. The old debate between top down and bottom up design became vivid again. The top down approach that is followed by ORACLE*CASE was difficult to grasp in particular by those who have a real time controls background. They felt more at home with the method and tools for real time application which tend to follow a more pragmatic middle out approach.

The entity relationship technique on which ORACLE*CASE is based led to some concern. NIAM on the other hand defines binary relationships between objects, and is felt by some as a more natural method for data modelling. It also provides diagrams that are more informative as they include explicit notations for e.g. role, subtypes and procedural constraints. A consequence of course is that NIAM diagrams are less surveyable than the ORACLE entity relationship ones. It is however possible to generate NIAM diagrams from ORACLE ones; the inverse operation is not possible because of loss of information.

The tools for real time application development highlighted the well known problem of transfer of information between the analysis and design phase. While it would seem that object oriented methods might provide an elegant solution to this problem, no tool has yet provided such a solution. The evolution of tools towards object orientation will thus need to be closely followed.

Because of the difference in version between ORACLE database and CASE tools (one version behind the database) the tables and forms that were generated were not making full use of the capabilities provided by the latest database version. A new version is scheduled for end of this year.

Several CASE tools exist on the market, each having their specific development area. In the absence of a fully integrated software engineering environment, one probably have to live with a variety of such tools. Customers would already be greatly helped if all tools could agree on a (set of) common repository (ies).

As a last and modest preliminary conclusion from the management point of view: with rather limited resources it was possible to introduce a reasonably large team into this, for CERN's accelerators at least, new technology; one can estimate the number of software engineers in the accelerator sector to date who are growing familiar with CASE, to around 30.

ACKNOWLEDGEMENTS

The authors are indebted to their colleagues at the European Southern Observatory, Garching-bei-München, Drs. G. Raffi and K. Wirenstrand, for valuable discussions and exchange of experience in the course of the evaluation of RTEE. They also benefitted from the professional advice and consultancy of Drs. J. Davis and L. Hickman both with ORACLE Corporation, U.K.

REFERENCES

[1] B.W. Boehm, Software Engineering Economics, 1981, p. 16.
[2] The Future of computed aided Programming, Corporate Communications 09/88 on "How will Technology affect America in the next 10 Years?", AEG 1988.
[3] B.W. Boehm, op. cit., p.118
[4] D.J.L. Martin, Practical Improvements in the Management of Real-Time Software Projects, 1st IFAC Workshop on Experience with the Management of Software Projects, May 14-16, 1986, Heidelberg, FRG.
[5] Computerworld, April 22, 1991, p. 76.
[6] Yourdon, Modern Sructured Analysis, 1989.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S15CSE02

Automation From Pictures: Producing Real Time Code from a State Transition Diagram*

Andrew J. Kozubal
Mail Stop H820
Los Alamos National Laboratory
Los Alamos, New Mexico 87545

Abstract

The state transition diagram (STD) model has been helpful in the design of real time software, especially with the emergence of graphical computer aided software engineering (CASE) tools. Nevertheless, the translation of the STD to real time code has in the past been primarily a manual task. At Los Alamos we have automated this process. The designer constructs the STD using a CASE tool (Cadre Teamwork) using a special notation for events and actions. A translator converts the STD into an intermediate state notation language (SNL), and this SNL is compiled directly into C code (a *state program)*. Execution of the state program is driven by external events, allowing multiple state programs to effectively share the resources of the host processor. Since the design and the code are tightly integrated through the CASE tool, the design and code never diverge, and we avoid design obsolescence. Furthermore, the CASE tool automates the production of formal technical documents from the graphic description encapsulated by the CASE tool.

I. INTRODUCTION

Structured analysis and design methods often make use of the state transition diagram (STD) to model real time systems.[1] A CASE tool, such as Cadre Teamwork/RT[2], can partially automate the STD methodology, but the programmer is left with the task of converting the STD into run time code. The programmer takes into account numerous factors, such as task priority, task synchronization, and pending for multiple events, to produce efficient code, and often the resulting code bears little resemblance to the STD. Using a two-step procedure, we have achieved significant automation of this process.

The translation of the STD into code is based on work done previously to develop a language that is based on the STD paradigm. The state notation language (SNL) [3] was developed to simplify programming of time-constrained sequential operations that are driven by events. During extensive experience with the SNL on the Ground Test Accelerator and the Advanced FEL at Los Alamos,[4,5] the SNL evolved into a powerful tool for implementing real time, automatic control. Subsequently, we developed a tool to capture relevant coding information about the STD within the CASE environment and translate it into SNL syntax. Below, we describe the salient features of the SNL, and explain how the translator is used to produce a complete SNL module from the STD.

II. STATE NOTATION LANGUAGE

We designed the SNL to be consistent with the STD methodology and applicable to the existing run time environment that we use at the Los Alamos National Laboratory.[6–8] Following the Mealy convention for STDs, we specify both the events and the actions on the transition between states, and allow only the state name to appear in the state as in Figure 1.

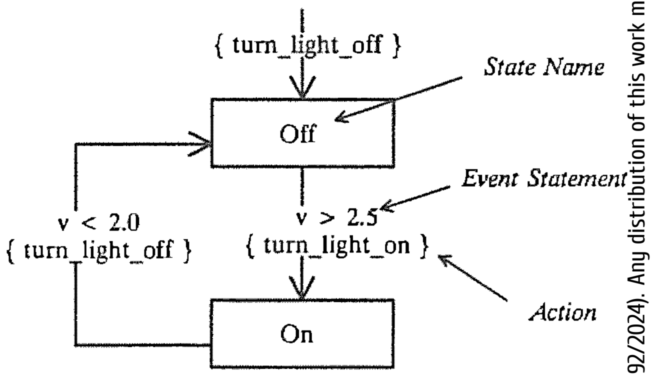


Figure 1. Example of a State Transition Diagram.

In the above example there is only a simple relational expression, which involves one event, the change in the value of variable “v”. The SNL is designed to handle more complex event expressions, as well as multiple events. Events may be associated with database channels and time delays. Actions may include calculations, outputs to database channels, and calls to procedures.

Rather than invent yet another new language, we based the SNL on a comprehensive subset of C, along with some relatively minor additions to handle events, actions, and states. We simplified the coding by allowing the programmer to associate run time database channels with a C variable. Figure 2 shows the complete program that implements the STD in Figure 1 in SNL syntax.

---

* Work supported and funded under the Department of Defense, US Army Strategic Defense Command, under the auspices of the Department of Energy.

```
program detect_HV_level;

float          v;
assign    v to "HV_PS_01:output_volts";
monitor   v;

short          light;
assign    light to "HV_PS_01:hv_light";

ss testHV {
     state Init {
          when () {
               light = 0;
               pvPut(light);
          } state Off
     }
     state Off {
          when (v > 2500.) {
               light = 1;
               pvPut(light);
          } state On
     }
     state On {
          when (v < 2000.) {
               light = 0;
               pvPut(light);
          } state Off
     }
}
```

Figure 2. A SNL Control Program.

A complete program contains a *program* statement, a declaration section, and one or more *state sets* (designated by "ss" in the SNL). Within a program, multiple state sets correspond to multiple STDs. Some of the SNL features include:

Statement	Description
program	Provides a name for run time execution.
assign	Assigns or associates a variable with a database channel.
monitor	Causes the channel value to be returned asynchronously whenever it changes by a significant amount.
ss	Specifies the start of a state set.
state	Specifies a state by name.
when	Specifies a transition, with the corresponding events. *When* is followed by the event and action statements and the next state.
pvPut	Function to put a value to a database channel.
pvGet	Function to get a value from a database channel.

The SNL is block structured, as in C. A state may have multiple *when* statements, corresponding to multiple transitions from that state. Other features of the language include: (1) macro definitions within database names, (2) network connection status of database channels, (3) access to channel alarm status, and (4) synchronization through event flags. A state notation compiler generates efficient reentrant C code from the SNL.

On the target processor a sequencer program initiates and controls the execution of a task for each state set. The sequencer establishes connections to database channels and handles asynchronous events, such as might occur on a monitored database channel or loss of a network connection.

III. INTEGRATING THE SNL INTO THE CASE TOOLSET

The user first builds a model within the Teamwork environment. By following existing conventions for real time analysis and design[9], the Teamwork will provide various checks on the design. The specification for a program begins at a process bubble within a data flow diagram (DFD). An example is show in Figure 3.

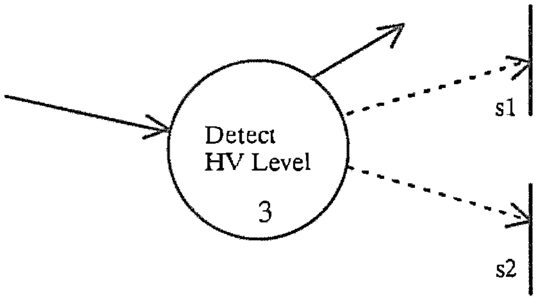


Figure 3. Part of a DFD Showing Control Connections to C-Specs.

Two "control flows" (dashed lines) from bubble 3 in this DFD connect to the control specifications (C-Specs), s1 and s2. Each C-Spec contains a STD, which corresponds to a state set in the program. Declarations, and other header information are placed in the process specification (P-Spec) that is contained within the DFD bubble. The events and actions are placed in the STD on the transitions. Because actions could be very complex — too many characters to fit conveniently on the STD — each action must be specified as the name of a P-Spec.

A translator builds the SNL program from the Teamwork model. This translator accesses the CASE model database using the Cadre Teamwork/Access interface routines[10]. To use the translator the user specifies the model name, the bubble number (default bubble is 0), and the output file for the SNL program. The topography of the DFD and STDs determine the program structure, and the contents of the STDs and P-Specs determine the details.

IV. EXPERIENCE AND FUTURE PLANS

We have used the translator on only a few simple test cases. The CASE tool methodology is a little awkward to use, especially when the programmer must go back and forth between the CASE environment and the run time environment during program debugging. On the other hand, programmers have been highly pleased with the SNL. We are investigating the idea of designing a graphic editor that would be more appropriate than the CASE environment.

Although we have made no measurements, we estimate that the use of the SNL rather than C has saved significant programming time, and that performance approaches that of programs written in C.

V. CONCLUSION

The STD paradigm is useful for implementing real time control. Automating the translation from STD to a run time program is expected to introduce fewer coding errors and provide better design documentation. Acceptance of this methodology may depend on providing a more user friendly graphic interface.

VI. REFERENCES

[1] Ward, Paul T. and Stephen J. Mellor, *Structured Development for Real Time Systems*, 3 vols., Prentice-Hall Inc., Yourdon Press, 1985.

[2] Teamwork/RT User's Guide, Release 4.0, Cadre Technologies Inc., Providence, RI.

[3] Kozubal, A. J., L. R. Dalesio, J. O. Hill, and D. M. Kerstiens, "A State Notation Language for Automatic Control," Los Alamos National Laboratory report LA-UR-89-3564, November, 1989.

[4] Atkins, W., "Using the EPICS Sequencer Tool to Automate the GTA Vacuum System, " submitted to Particle Accelerator Conference, San Francisco, California, May 6—9, 1991.

[5] Wilson, William L., Marcus W. May, and Andrew J. Kozubal, "Rapid Development of a Measurement and Control System for the Advanced Free-Electron Laser," submitted to Thirteenth International Free-Electron Laser Conference, Santa Fe, New Mexico, August 26—30, 1991.

[6] Kozubal, A. J., L. R. Dalesio, J. O. Hill, and D. M. Kerstiens, "Run Time Environment and Applications Tools for the Ground Test Accelerator Control System," in *Accelerator and Large Experimental Physics Control Systems*, D. P. Gurd and M. Crowley-Milling, Eds. (ICALEPCS, Vancouver, British Columbia, Canada, 1989), pp. 288—291.

[7] Dalesio, L. R., "The Ground Test Accelerator Database: A Generic Instrumentation Interface," in *Accelerator and Large Experimental Physics Control Systems*, D. P. Gurd and M. Crowley-Milling, Eds. (ICALEPCS, Vancouver, British Columbia, Canada, 1989), pp. 288—291.

[8] Hill, J. O., "Channel Access: A Software Bus for the LAACS," in *Accelerator and Large Experimental Physics Control Systems*, D. P. Gurd and M. Crowley-Milling, Eds. (ICALEPCS, Vancouver, British Columbia, Canada, 1989), pp. 288—291.

[9] Hatley, Derek J. and Imiaz A. Pirbhai,, *Strategies for Real Time System Specification,* Dorset House Publishing Co., 1987.

[10] Teamwork/Access, Teamwork/RT User's Guide, Release 4.0, Cadre Technologies Inc., Providence, RI.

SSC Lattice Database and Graphical Interface

C. G. Trahern and J. Zhou
Superconducting Super Collider Laboratory
2550 Beckleymeade Ave., Dallas, Texas 75237*

Abstract

The SSC lattice database and the graphical tools used to access it are discussed.

I. INTRODUCTION

When completed the Superconducting Super Collider will be the world's largest accelerator complex. In order to build this system on schedule, the use of database technologies will be essential. In this paper we discuss one of the database efforts underway at the SSC, the lattice database. The original work on this database system began at the SSC Central Design Group and is described in reference [1].

The SSC lattice database provides a centralized source for the design of each major component of the accelerator complex. This includes the two collider rings (top and bottom), the High Energy Booster (HEB), Medium Energy Booster (MEB), Low Energy Booster (LEB) and the LINAC as well as transfer and test beam lines.

These designs have been created using a menagerie of programs such as SYNCH, DIMAD, MAD, TRANSPORT, MAGIC, TRACE3D and TEAPOT. However, once a design has been completed, it is entered into a uniform database schema in the database system.

In section II we further discuss the reasons for creating the lattice database and its implementation via the commercial database system SYBASE[2].

Each lattice in the lattice database is composed of a set of tables whose data structure can describe any of the SSC accelerator lattices. This data structure will be discussed in section III.

In order to allow the user community access to the databases, a programmatic interface known as dbsf (for database to several formats) has been written. This interface is the subject of section IV. dbsf creates ascii input files appropriate to the above mentioned accelerator design programs. In addition it has a binary dataset output using the SDS (Self Describing Standard) data discipline provided with the ISTK (Integrated Scientific Tool Kit)[3] software tools.

In section V we discuss the graphical interfaces to the lattice database. The primary interface, known as OZ, is a simulation environment as well as a database browser.

OZ has been created using techniques of object oriented modelling and coded in C++ using the ISTK software tools. OZ is SSC specific in that it presents a geometrical view of the collider complex from which one may select the lattice of interest. This geometrical data is also the source of the data used to site the complex geographically. OZ is an interactive simulation environment in which the user can change various parameters such as a steering magnet's field strength and then see whether a beam of given emittance will survive its propagation within a fixed spatial aperture.

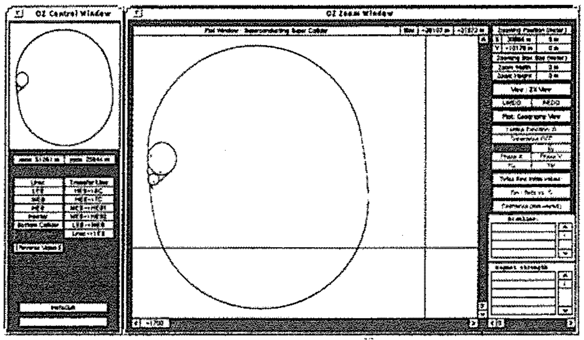


Figure 1: Control and Display Windows of OZ

In addition to the geometrical view of the complex, one also needs a more abstract view of a lattice structure, and this is provided by a program known as LATVIEW. Because the beam line hierarchy implicit in an accelerator design can be highly nonintuitive, LATVIEW gives an interactive graphical view of this hierarchy.

II. PHILOSOPHY

We have implemented the lattice database using a relational database management system (RDBMS). The particular software system currently used is SYBASE operating within a UNIX workstation computing environment. By putting the lattice information within a RDBMS tied to a network, essentially universal access to the data can be supported. In addition by maintaining a uniform description of the lattice information, various groups such as mechanical and civil engineering, survey and alignment as well as diagnostics and simulation can be coupled to the same data in an efficient manner thus reducing the probability that different groups will use incompatible design information.

In addition, because accelerator design is usually done with a variety of design codes such as the ones mentioned earlier, a lattice database of some kind is the only way to

---

*Operated by the Universities Research Association, Inc., for the U.S. Department of Energy under Contract No. DE-AC35-89ER40486.

effectively manage the design process. The fact that our choice has been to use an RDBMS reflects the desire to learn new software technologies as well as to avoid the paper trail of past accelerator construction experience. The lattice database effort should be regarded as one of the first experiments at the SSC in using new technical organizational schemes.

SYBASE is one of the major SQL (structured query language) RDBM systems available. It runs on a variety of hardware platforms, and has an excellent network interface. Lattice database applications have been written using C and the Open Client/C software library supplied by SYBASE. The performance of these applications has been exceptional. In the future the applications should be written with embedded SQL so that other users at different sites should be able to use the lattice database system with another RDBMS.

III. LATTICE DATABASE

Lattice Structure as Database Tables

In creating the lattice data structure, we have adopted the MAD 8.1 element definitions[4] since MAD provides a *de facto* standard for accelerator design descriptions. Each lattice is assigned its own database in the RDBMS with the same set of generic tables in each database.

A generic lattice database is made of five primary tables: beam_line, slot, magnet_piece, geometry, strength, and 12 secondary tables: quadrupole, sextupole, octupole, multipole, rfcavity, bend, drift, collimator, closed_orbit_corrector, monitor, elseparator and solenoid. The primary tables define a basic lattice hierarchy of beam lines, components and parameters and the secondary tables provide additional information which completes the MAD 8.1 element definitions.

The beam_line and slot tables define the beam line hierarchy. Drifts, magnets or other elements are entered into 'slots' as sequences of character strings by their names and slots are strung together in other character strings to make beam lines. Beam lines can also be concatenated into larger beam line structures and so on.

The magnet_piece table contains columns which describe the basic physical properties of each element such as its type, length, strength, etc. The geometry and strength tables store physical parameter names and values such as lengths and magnetic properties such as field values or field gradients. Two tables were created for parameters instead of one because there will certainly be different versions of the strength table associated with different operating conditions of an accelerator such as injection, collision and points in between. The application program can be directed to choose the strength table of choice.

In our experience to date the MAD element definitions have been appropriate for all accelerator lattices excepting the linac. In order to provide a faithful database representation of the linac design, we have had to implement an additional set of tables which define new lattice elements needed by TRACE3D, a linac design program in general use at the SSC.

In principle it is now straightforward to adapt to new demands on the data structure by creating new tables when necessary. For example, in order to describe the layout of the assembly of magnets to determine the existence of physical interferences, information concerning the outside dimensions of magnet elements is needed in the database. This information is maintained in a table known as magnet_size which contains columns detailing the shape and external dimensions of a given element. The primary key of this table (and most others) is the element's name. Breaking this information off into another table rather than adding a new column to the magnet_piece table for example, helps to separate applications which are independent of each other. For example a beam line survey program which models physical interferences may require information about the outside dimensions of beam line components, but does not need the magnet strength values appropriate for linear optics calculations.

SQL example

An example of SQL code which creates the magnet_piece table:

```
create table magnet_piece
(name             char(20),
type              varchar(20),
tilt              varchar(10)    null,
length_defn       varchar(60)    null,
strength_defn     varchar(60)    null,
engineering_type  varchar(20)    null,
comment           varchar(130)   null)
```

The first column above specifies the table's column names. They are name, type, tilt, length_defn, strength_defn, engineering_type and comment. Name is the ascii name of the beam line component. Type refers to one of the MAD element types such as drift, quadrupole, sextupole, etc. Tilt specifies the orientation of the magnetic element around the beam line. The meaning of strength_defn can depend on the type. For example, the quadrupole strength is proportional to the gradient of the magnetic field, and the sextupole strength is proportional to the second partial derivative of the magnetic field strength. Symbolic algebraic expressions for these quantities can be entered in this column as long as the parameters used are defined elsewhere in the geometry or strength tables. Engineering_type is an alternative name which the user may introduce to flag components for special purposes in the application codes.

The second column above specifies the data types. For this table these data types are 'char' and 'varchar'. These character field types differ from each other in that char uses all of its allocated space and varchar only stores that portion of the allotted space of the field which is actually used. Finally, the null qualifier on some columns means that no data is required in that column; both name and type being non-null are required to be entered by the database user. The other database tables have similar structure, and all tables have a unique index defined on the primary key, usually the first column. These indices help maintain the integrity of a table by prohibiting multiple entries with the same primary key.

IV. USER INTERFACE to DATABASE

dbsf (database to several formats) is the primary interface to the lattice database. It is written in C and uses the Open Client/C libraries provided by SYBASE. To run dbsf one supplies any lattice keyword such as a beam line, component or parameter name. dbsf first performs a query to determine the location by table of this keyword, and then searches that table for information about the key. If other keys are associated with the primary key, these are also made the subject of similar database queries until no further unknown keys are encountered. At this point an extended symbol tree has been constructed within dbsf and depending on one of many options selected at the UNIX command level, dbsf will format an input file appropriate to that option. The first options available were for MAD. Since that time, all the accelerator codes listed in the introduction are supported.

Recently, an option to create a binary dataset using the SDS data discipline has been installed. SDS is provided as part of the ISTK tool set. The SDS dataset binds a description of the data structure used to create the data within the dataset itself as a header object. Consequently, within the context of database applications, SDS data can maintain the integrity of the database structure outside of the RDBMS. Application programs that read generic SDS data can then be used with this data. In particular, SDS data can be passed from one RDBMS to another through ISTK supplied interfaces. Such interfaces can provide a simultaneous solution to the problems of making heterogeneous database system communicate effectively and also allow large data to flow quickly from one point on a network to another. Several programs which are specific to lattice issues will be discussed below.

V. GRAPHICAL INTERFACES

OZ

Early in the development of the lattice database, the design of a graphical interactive view of the complex of accelerators was begun. The name of this program is OZ. OZ was required to display a geometrical view of the entire accelerator complex and to have the ability to use a 'mouse' to select a part of the accelerator system and see that area visually expanded in greater detail while displaying the relevant properties of the selected beam line components. In addition the interface was required to provide a basic simulation environment for each of the accelerator lattices. Consequently, it should not only display the design values of linear optical properties (betatron functions, dispersion) correlated with selected regions of a lattice, but also allow the user to modify the magnetic strengths and recompute the optics.

Finally, the interface should include a particle tracking module. The tracking module would use a fixed initial emittance profile to define positions and momenta of a small number of particles, and then propagate these particles and display the resulting emittance profile. This is done after one turn for circular lattices and at the end for transfer lines. The stated goal of the particle tracking module was to provide the user with a convenient way to modify steering elements strengths and see the result graphically. In this way one could study the effects of beam apertures quickly. The tracking module was not intended to model the various errors in magnetic field strength or alignment. It provides an ideal view of the particles' motion.

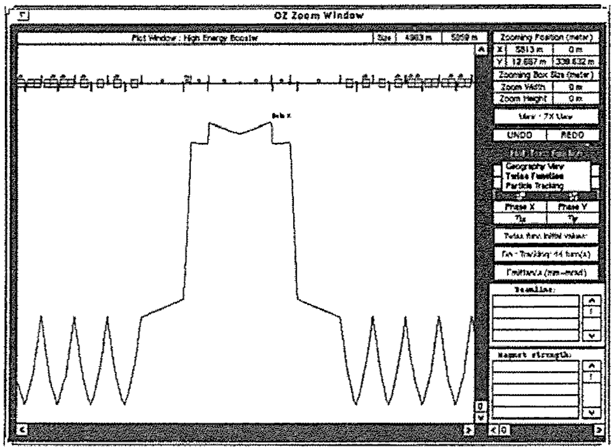


Figure 2: Betatron Function Window in OZ

As the general computing environment of the SSC Accelerator Division is based around UNIX workstations running the X11 windowing system, the graphical tools used to build OZ were chosen to be compatible with this environment. The original development was begun using the InterViews 2.6 toolkit[5], but was later changed to the graphics libraries provided with ISTK. These graphics tools, known as glistks, are based on a subset of InterViews' Interactor class and are tailored to build scientific graphical interfaces.

The interactive capability of OZ is manifold. After selecting one of the lattices from the command window, OZ displays the geometrical view of that lattice in the display window. One can select a portion of that lattice for expansion by grabbing the region with the mouse. One menu then allows you to view that lattice from different two dimensional perspectives, or to query the database to find a particular beam line name. Another menu allows the user to view the linear optical properties. A one dimensional version of the lattice is displayed at the top of this view so that the user can tie the optics to element locations.

Finally, one can select the tracking menu in order to define initial parameters for particle tracking such as

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S16MS01

coordinates and momenta or change the desired emittance profile. OZ will then track the particles for a few turns and display the final profile in a series of windows showing horizontal and vertical phase space as well as the purely spatial transverse profile of the beam.

LATVIEW

Another graphical interface to the lattice database is called LATVIEW. This program was designed to allow visualization of the lattice hierarchy. A lattice such as the main collider ring of the SSC is composed of more than 20,000 elements including drift spaces, magnets, beam position monitors and position markers. The collection of beam lines which combine these elements into a lattice design is sufficiently complicated that a graphical interface to its structure is absolutely necessary for most users.

LATVIEW uses the SDS output from dbsf as its source of information for display. This SDS dataset includes as one of its elements the enumeration of the 'level' at which a beam line exists in the hierarchy. The notion of level is defined so that level 0 corresponds to a completely flat lattice, i.e., with all trace of the beam line hierarchy eliminated. Levels 1 and 2 show the relationship of the basic elements to the slots in which they belong, and levels 3 and so on define the tree of nested beam lines whose highest level corresponds to the complete lattice's name, the original keyword given to dbsf needed to generate the dataset.

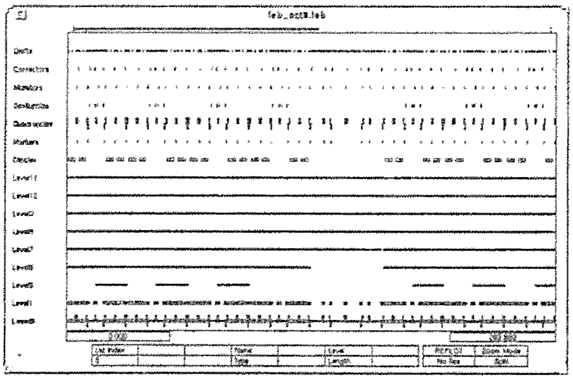


Figure 3: Lattice Hierarchy of LEB from LATVIEW

LATVIEW displays information on each element (beam line or individual magnet) by placing the mouse on the element of interest. This action provides sufficient information to point back within the SDS dataset and pick out the relevant information pertaining to that element. The information is displayed at the bottom of the LATVIEW window. One can also select a part of the lattice with the mouse, and LATVIEW will show the expanded view. In addition to its display of the beam line hierarchy, LATVIEW will also display groups of magnets by type. So, for example, if one is interested in locating all quadrupole locations along the beam line, LATVIEW provides a convenient way to do this visually without losing the relationships of these magnets to their parent beam lines.

VI. CONCLUSIONS

The SSC lattice database has been in existence for two years. It has been an essential part of ongoing operational simulation investigations as well as being the basis for engineering drawings of each accelerator system and the geographical footprint of the entire complex of accelerators. As the SSC moves into the construction phase, additional database structures will be needed to describe the hierarchy of cryogenic, electrical and control systems that overlay the basic lattice designs. In addition the specification of an automated storage/retrieval system for the names and locations of the multifarious pieces of equipment that are attached or associated in some way with the lattice complex is essential. Some of these problems are under investigation, and present efforts are directed at the determination of requirements for these global database systems.

References

[1] E. Barr, S. Peggs, and C. Saltmarsh, SSC-N-606, March 1989.

[2] SYBASE, Inc., 6475 Christie Ave., Emeryville, California, 94608.

[3] C. Saltmarsh, M. Allen and S. Acharya, ISTK, unpublished internal documentation at SSCL, July 1991.

[4] D.C. Carey and F. Christopher Iselin, Proc. of the SSC Summer Study, p. 389, Snowmass, 1984 and The MAD Program, F. Christopher Iselin, CERN/SL/90-13 (AP).

[5] Mark A. Linton, InterViews Reference Manual, v. 2.6, February 1990, Stanford University, Stanford, California.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S16MS02

Framework for Control System Development*

Carl Cork and Hiroshi Nishimura
Lawrence Berkeley Laboratory, 1 Cyclotron Road, Berkeley, CA 94720

Abstract

Control systems being developed for the present generation of accelerators will need to adapt to changing machine and operating state conditions. Such systems must also be capable of evolving over the life of the accelerator operation. In this paper we present a framework for the development of adaptive control systems.

I. INTRODUCTION

Several of the new generation of control systems hardware being developed today have the capability of fast, sophisticated control at all levels in the control hierarchy[1][2]. These systems are typically hierarchical and highly distributed with extremely high I/O throughput.

We have initiated the design of a framework for control system development which can accommodate the new architectures. This paper will present requirements, design decisions, and specifications that we have devised for this framework.

II. REQUIREMENTS

A. *Adaptive*

The control system must be adaptive. It must be capable of growth, evolution, and learning (supervised and self-taught).

The software for these systems is complex and generally in continuous development. The control system must be capable of growth during both commissioning and operational phases.

Many new control system algorithms such as model-based control, expert systems, neural networks, and fuzzy logic are emerging which look very promising in the accelerator control environment.[3][4][5][6]. A mechanism is required which is capable of evolution to accommodate these new control theories. The system must also be capable of arbitrarily complex combinations of these algorithms.

Most of these new control system algorithms are capable of either supervised or self-taught learning. This should prove to be extremely useful as an aide to finding 'golden *orbits*' in storage rings or as a means of reducing the complexity of data presented to the operator. The control system must facilitate this mechanism.

B. *Hierarchical*

The control system must support a hierarchical control structure. It must be capable not only of supporting the 'standard' supervisor-cell-local type of hierarchical control[7], but also each layer must be divisible into local subhierarchies. This latter requirement facilitates the incorporation of cascaded and adaptive control algorithms.

C. *Distributed*

The control system must support the underlying distributed hardware.

Many computer systems provide basic networking support. The control system must also incorporate mechanisms for the registration of computing services, the automated association of client and server, and the uniform representation of data transmitted between heterogeneous systems.

The control system must be designed to accommodate the known features of distributed control - such as error detection and recovery, virtual time synchronization, nondeterministic networks, concurrency, resource protection, and bandwidth-limited messaging.

D. *Operational Continuity*

The control system must support operational continuity. It must provide for dynamic, and transparent switching between compatible modules without interrupting operation.

Transparent switching is required to permit the exchange of control modules in the event where the system operation exceeds the bounds of the previous controller. This should be possible without bringing the system down and without leaving the machine uncontrolled. Sufficient machine state information should be transferrable to provide for 'bumpless' switching.

E. *Dynamic Association*

The control system must support the dynamic association of applications. Links between the control system and the application should be redirectable during normal operations. This is essential to provide for independent development of associated modules and also to provide support for the adaptive and operational continuity requirements listed above.

Dynamic association permits both application and control modules to be constructed without prior availability of the associated modules. Moreover, for client-server associations, the link process should not require specific knowledge of the server module (capability-based binding). It should be sufficient to specify the type of module and its interface, leaving the association mechanism to a third intermediate process.

F. *Universal Graphical API*

The control system must support a universal graphical application programming interface (API). Regardless of the operating system, windowing system, or window manager, the graphical application programming interface should be identi-

---

* This work was supported by the US DOE under Contract No. DE-AC03-76SF00098

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S16MS02

cal. All that should be required is a recompilation for each graphical display workstation.

III. DESIGN

Based on the requirements listed in the previous section we have established the following design specifications.

A. *Virtual Control Modules*

The control system shall accommodate the adaptability, hierarchy, and continuity requirements by incorporating a recursive architecture. This may be expressed using the modified Backus-Naur Form (BNF) formalism which is often used to specify computer language syntax:

$$VirtualControlModule = ControlModule + [VirtualMachine]^{+} \qquad (1)$$

$$VirtualMachine = Machine \mid VirtualControlModule \qquad (2)$$

The *VirtualControlModule* is the control subsystem consisting of a controller (*ControlModule*) and one or more controlled objects (*VirtualMachine*). The *VirtualMachine* may consist either of the bare *Machine* or, recursively, of an additional *VirtualControlModule* subsystem.

The *Machine* represents the accelerator and its associated instrumentation. This system may be represented by a set of n measures (state variables) and its development over time may be expressed by trajectories of the state variables in state space. During the development and testing phases, the *Machine* might be replaced by a simulator which emulates all command and response characteristics of the real machine.

The *ControlModule* is required to counteract any motion of the machine system away form the stable operating point. The combined subsystem (*ControlModule + VirtualMachine*) should be asymptotically stable. In the adaptive control system the *ControlModule* is a mutable element. Its state parameters are dynamically adjustable, it might be layered, parallelized, or self-adaptive. It shall also be dynamically replaceable by an alternative *ControlModule* with synchronized exchange of control between the *ControlModules*.

This model can describe all of the standard control systems in use today. The following expressions represent a few such systems.

$$\begin{aligned} RemoteControlSystem &= ControlModule + Machine \\ SupervisedControl &= ControlModule \\ &\quad + \langle ControlModule + Machine \rangle \qquad (3) \end{aligned}$$

A control system which accommodates *VirtualControlModules* will permit the control system to be modularly adjustable and to incorporate growth (and scalability) and evolution.

B. *Distributed Task Synchronization*

A synchronization mechanism shall exist to coordinate interaction with the machine elements and peer subsystems. The distributed machines shall incorporate partially ordered logical clocks to support virtual synchronization between coordinating processes across the network[8][9].

A multitasking environment shall be incorporated to provide synchronous and concurrent behavior on a single system. Task synchronization can be performed using any of the typical real-time mechanisms (e.g. semaphores, message queues, mailboxes).

C. *Distributed Computing Services*

A peer-to-peer message passing mechanism shall be implemented to satisfy the distributed communications requirements. This mechanism should have a programming interface which is independent of the network transport layer implementation. The design should be efficient enough to consider using it equally for local or remote task-to-task communications.

The control system should also support both message-based and remote procedure-based communication mechanisms. Message-based mechanisms will probably be best suited for event-driven processes which would normally be looping on an input message queue. Remote procedure-based communications will be best suited to transparent migration of library modules from local to remote configurations.

D. *Object Communications Manager*

An object communications manager shall be implemented to satisfy the dynamic task association requirement. The object communications manager will coordinate the interaction between applications and all other elements of the control system. The control system elements will be composed of software objects which interact to perform their assigned functions. Some of these objects will "advertise" their presence to external applications by registering with the object manager. External applications will query the object manager to select and associate with the advertised interfaces. The object communication manager permits the association to be dynamic and transparent. New control system objects can be substituted without requiring a restart of either the user applications or the control system modules. Moreover, the user application need not know whether or not the control system modules are operating locally or remotely - the interface is the same for both (the mechanism is similar to the X-windows byte-stream implementation). The operator interface applications are specific examples of applications which will use the object communication manager to interact with the control system.

All potential *ControlModule* and *VirtualMachine* modules must satisfy uniform interface requirements with respect to the object communications manager. This permits the modules to be dynamically replaced during operation and without requiring the reconfiguration of existing modules.

E. *Network-Based GUI*

A network-based graphical user interface (GUI) shall be incorporated to satisfy both the universal graphics application programming interface and the distributed control requirements. The graphical display will be presentable on any candidate workstation on the network. The display application may reside either within the workstation or else on some remote computer. The interface must function in a heterogeneous environment and should function on a variety of platforms.

IV. FRAMEWORK

The following framework was established to implement the design specifications from the previous section. The framework is based on existing technology and/or standards. This was done not only to take advantage of commercial products and community efforts but also to guarantee a more timely implementation of the composite system. We attempted to select a minimal framework to avoid an overly restrictive development environment.

A. *Eiffel Object-Oriented Environment*

The *Eiffel* object-oriented programming language and application environment will be implemented to satisfy the virtual control module design specification[10].

Object-oriented environments support modular software development, data abstraction, polymorphism, and dynamic binding - all of which are required to satisfy the virtual control module specification. *Eiffel* in particular also supports automatic memory management, multiple inheritance, enhanced reusability, and a special reliability feature (assertions) which supports a 'software by contract' design methodology. The language specification is now in the public domain and has a strong international and educational backing which should assure its continual evolution.

B. *POSIX 1003.1 Operating System*

The standard multitasking operating system will be a real-time operating system which is compliant with the IEEE specifications for a portable operating system interface (POSIX 1003.1). It will also support the real-time extensions (IEEE 1003.4) which are presently awaiting finalization.

At the higher machine architecture level we will select either the LynxOS or else the Chorus real-time operating system[11][12]. Both of these are network-based, POSIX compliant, and support real-time computing features.

At the lower machine control level we will use the VxWorks operating system[13]. This is a network-based, embeddable real-time operating system with a wide support base in the VME environment. VxWorks will provide a POSIX compliant interface when the real-time extensions are finalized.

C. *Distributed Computing Environment*

The distributed computing environment (*DCE*) will be implemented using the *OSF/DCE* utilities from the Open Software Foundation (*OSF*)[14]. These utilities will provide basic services for remote procedure calls, network security, and distributed file systems. The *OSF/DCE* is layered upon any POSIX compliant interface and is composed of elements which are available commercially today.

These utilities will soon be available on all major variants of the UNIX operating system. Initially, it will be available from *OSF* on their POSIX compliant operating system, and later it will be available from the Unix Software Laboratory (USL) on their SVR4 UNIX base. The Open Network Computing (ONC) utility set which is the dominant remote procedure call facility in use today will probably adapt to incorporate *DCE* compatibility.

D. *Object Request Broker*

The object communication manager facility will be provided by the Object Request Broker (*OMG/ORB*) which is being specified by the Object Management Group in collaboration with several large computer companies[15]. Early versions of this facility will be available from Hewlett-Packard and from Sun Microsystems.

A working example of this facility, ToolTalk, is currently available from Sun Microsystems for use on their workstations[16]. Our first *ORB* compliant applications will probably be based on this toolkit.

E. *X-Windows, Motif, and IEEE 1201.1*

The network-based graphical user interface will be provided by the MIT X-Window system, the *Motif* graphical user interface, and the evolving IEEE 1201.1 universal application programming interface libraries.

The only universal, network-based, window environment available today is the X-Window system. The latest release (X11R5) is fast, supports scalable fonts, and runs on every major UNIX workstation. A large amount of public domain software is available for this windowing environment.

Unfortunately, there are several competing, incompatible, graphical user interfaces available for the X-Window system. The *OpenLook* GUI is being promoted by AT&T and Sun Microsystems, while the *Motif* GUI is being promoted by the Open Software Foundation (OSF) and most of the other workstation vendors. However, to our knowledge the *Motif* window manager and application programming interface is also the only environment which runs universally on all present Posix compliant systems. Moreover, a number of *Motif* compliant GUI tools are available for most of these platforms.

The IEEE 1201.1 committee is developing a specification for a standard GUI programming interface which can be used with any of the X-Window GUIs in use today. We will adopt this standard when it becomes available, but in the meantime we will use the Xm-based toolkit from *OSF* for their *Motif* GUI. Wherever possible, we will also be using the *Eiffel*-based graphics toolkit from Interactive Software Engineering[17].

V. PROJECT STATUS

We have developed several prototype components to test some elements of this framework.

A class library for accelerator modeling and simulation has been constructed using *Eiffel*. Another *Eiffel* class library for the hardware database access is also being developed which interfaces with the LBL/ALS control system. Using these class libraries, one can create accelerator models dynamically with on-line and real-time access.

Several network-based, object-oriented device handlers have been written running under the VxWorks operating system on a VME target system. These handlers are being rewritten in *Eiffel*. An object management broker for VxWorks is also in progress.

VI. REFERENCES

[1] S.A. Lewis, A.K.Biocca, R.D.Dwinell, J.R.Guggemos, L.L.Shalz, W.L.Brown, G.S.Boyle, K.Fowler, and D.L.Meany, Progress on a New Control System for the Bevelac, IEEE Part. Accel. Conf., Vol. 89CH2669-0(1989)1645.

[2] S.Magyary, M.Chin, C.Cork, M.Fahmie, H.Lancaster, P.Molinary, A.Ritchie, A.Robb, and C.Timossi, Advanced Light Source Control System, IEEE Part. Accel. Conf., Vol. 89CH2669-0(1989)74.

[3] M.Lee, S.Clearwater, E.Thiel, and V.Paxson, Modern Approaches to Accelerator Simulation and On-Line Control, IEEE Part. Accel. Conf., Vol. 87CH2387-9(1987)611.

[4] S.H.Clearwater, and M.Lee, Prototype Development of a Beam Line Expert System, IEEE Part. Accel. Conf., Vol. 87CH2387-9(1987)532.

[5] J.E.Spencer, Accelerator Diagnosis and Control by Neural Nets, IEEE Part. Accel. Conf., Vol. 89CH2669-0 (1989) 1642.

[6] Bart Kosko, Neural Networks and Fuzzy Systems. A Dynamical Systems Approach to Machine Intelligence, Prentice Hall, Englewood Cliffs, NJ, 1992.

[7] B.Lipták, and K.Venzzel, Instrument Engineers' Handbook, Process Control, Chilton, Radnor, PA, (1985) 713.

[8] C.Fidge, Logical Time in Distributed Computing Systems, IEEE Computer, Vol.24,No.8(1991)28.

[9] Isis-A Distributed Programming Environment, Version 2.0 user's guide and reference manual, Cornell University (April 1990).

[10] Bertrand Meyer, Object-Oriented Software Construction, Prentice Hall, Englewood Cliffs, NJ, 1988.

[11] LynxOS, Lynx Real-Time Systems, Inc., Los Gatos, CA, USA.

[12] CHORUS/MIX, Chorus Systems, Beaverton, OR, USA.

[13] VxWorks, Wind River Systems, Inc., Alameda, CA, USA.

[14] Distributed Computing Environment, Open Software Foundation, Cambridge, MA, USA.

[15] Object Request Broker, Object Management Group, San Francisco, CA, USA.

[16] ToolTalk (Beta) Programmer's Guide, Part No: 800-6093-05, Sun Microsystems, Inc., Mountain View, CA, USA.

[17] Eiffel Graphics Library, Interactive Software Engineering, Inc., Goleta, CA, USA.

The LEP Model Interface for MAD

F. Christoph Iselin
CERN, SL Div.
CH-1211 Geneva 23
Switzerland

Abstract

During machine studies and trouble-shooting in the LEP machine various optical parameters must be computed, which can be found quickly using the MAD program. However, the LEP operators are not all well acquainted with MAD. In order to ease their task, a simple interface called the LEP model has been written to run on the Apollo workstations of the LEP control system. It prepares jobs for MAD, sends them to a DN 10000 node for execution, and optionally plots the results.

The desired machine positions and optical parameters vary between LEP runs. The LEP model contains a powerful selection algorithm which permits easy reference to any combination of positions and optical parameters in the machine. Elements can be chosen by name, by sequence number, or by element class. The choice of optical functions includes closed orbit, Twiss parameters, betatron phases, chromatic functions, element excitations, and many more. Recently matching features have been added.

Communication with the control system and with MAD uses self-describing tables, i. e. tables whose columns are labelled with their name and a format code. Experience with this LEP model interface is reported.

1 Introduction

This section describes those aspects of the LEP control system and of MAD which are relevant to the LEP model program. The second section outlines features of the LEP model program. The third section discusses implementation, and the last two sections present future plans and experiences with the program.

1.1 LEP as seen from LEP Model

The LEP control system [1] is based on a network of Apollo workstations connected in a token ring network. The workstations are running under UNIX. They talk to the LEP machine over various links and microprocessors. For time-intensive tasks the network contains an Apollo DN 10000 computer, whose speed is about a factor 1/2 of the IBM 3090.

The descriptions of the LEP machine and of its possible optical configurations reside in an Oracle database. From the database a structural description of LEP is available which is formatted in MAD input language.

For equipment control the access to the Oracle database is too slow. A set of files, known as the "reference data set", is thus extracted and stored in a file server. Most of these files are self-describing tables, known as TFS tables (Table File System [5]). Each table has an arbitrary number of descriptors, and each column is labelled with its name and format code.

The status of equipment, e. g. the magnet excitations, or the RF cavity settings, can be acquired via specialized programs and is usually stored in TFS format. TFS tables can also be sent to LEP to modify the settings of equipment.

1.2 MAD seen from LEP Model

The MAD program [2, 3] has been used extensively for the design of LEP. It is based on a "standard language" [4], used to describe the machine structure, and to request various computations on this structure. The language is designed to make communication with a human user easy. For communication with other programs MAD also understands TFS format.

In the framework of the LEP Model Program MAD serves the following purposes:

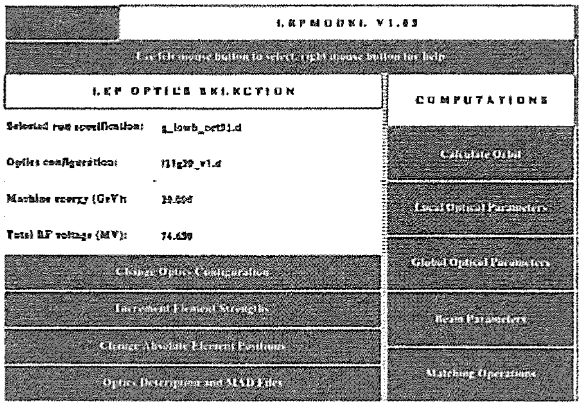
- Compute the closed orbit,
- Compute optical functions over parts of the machine,
- Match optical functions to specific conditions,
- Calculate global parameters of LEP,
- Change machine parameters to study their effect.

2 The LEP Model Program

2.1 Tasks

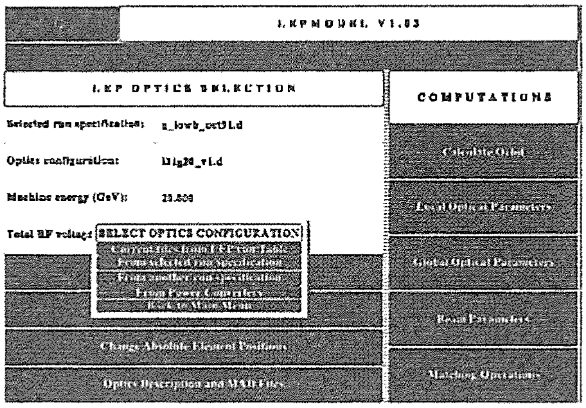
Based on the above, the LEP Model Program must

- Use the reference data set to build menus of available optical configurations and to present them to the user for choice. In this way the program needs no changes if new configurations are installed. The proposed default is taken from a file known as the LEP Run-Table.



Selected run specification: s_lowb_oct31.d
Optics configuration: l11g29_v1.d
Machine energy (GeV): 20.000
Total RF voltage (MV): 74.630

Figure 1: Opening Screen for LEP Model Program



Selected run specification: s_lowb_oct31.d
Optics configuration: l11g29_v1.d
Machine energy (GeV): 20.000
Total RF voltage:

Figure 2: Menu to Change Optical Configuration

- Deliver the following files to MAD:

  1. The definition of elements (magnets, cavities, etc.) in the machine, and the sequence of their occurrences. This file is built in the Oracle database and formatted in the standard MAD input language. It only changes when the LEP machine is modified physically.
  2. The current optical configuration (magnet excitations). This file is a TFS table selected from the reference data set, or built by reading the actual power converters by launching a specialized program.
  3. The "imperfection file". This file contains known imperfections of the machine, like measured multipole components in the machine dipoles. It is updated manually when such imperfections become known.
  4. The "trim file" introduces factors to compensate observed errors of the model. The corrections are constructed by trial and error, and do not necessarily represent actual imperfections of LEP.

- Allow interactive increments of element excitations.

- Allow interactive assignment of element displacements.

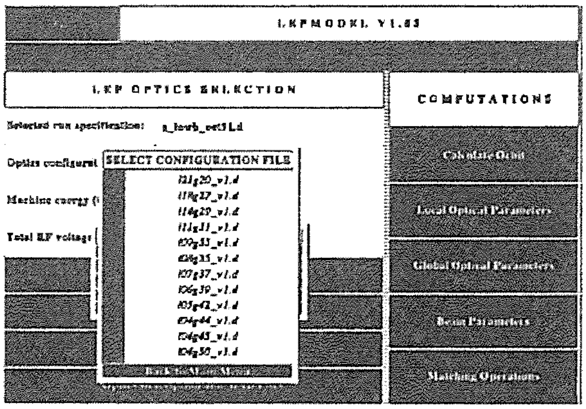
- Set up a MAD command file to read the above files, and to perform the desired computations.

- Launch MAD on the DN 10000 server.

- Present a table of results on the display.

- Optionally plot the results on the display.

In the standard version of the LEP Model Program all operations must be done in such a way as to avoid MAD to fail during computation.



Selected run specification: s_lowb_oct31.d
Optics configurat
Machine energy (
Total RF voltage

Figure 3: Menu of Available Optical Configurations

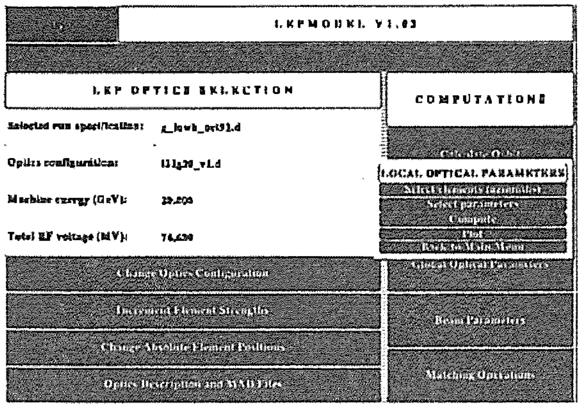
2.2 Examples of Menu Selections

The opening screen of the LEP Model Program is shown in Figure 1. Selection of the "Change Optics Configuration" icon pops the menu shown in Figure 2. It presents options for selecting configuration files. Selection of "From current run specification" then presents the menu of Figure 3, containing all states for the current run. To compute the optical parameters in selected positions the user first selects "Local Optical Parameters" from the main menu and gets the menu shown in Figure 4 After selection of "Select Elements" the program pops the menu shown in Figure 5 which allows to select the positions where to calculate the parameters. Selection of "Select Parameters" then presents the menu of Figure 6 offering the available optical functions. While it is computing, the LEP Model Program displays a screen like in Figure 7. The results may be plotted as shown in Figure 8.

3 Implementation

3.1 Choice of User Interface Package

Domain/Dialogue was chosen for the following reasons:



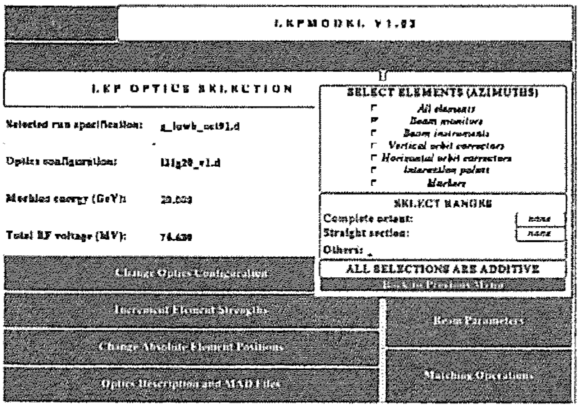
Selected run specification: g_lowb_oct91.d

Optics configuration: l11g20_v1.d

Machine energy (GeV): 20.000

Total RF voltage (MV): 74.430

Figure 4: Menu for Local Optical Parameters



Selected run specification: g_lowb_oct91.d

Optics configuration: l11g20_v1.d

Machine energy (GeV): 20.000

Total RF voltage (MV): 74.430

☐ All elements
☑ Beam monitors
☐ Beam instruments
☐ Vertical orbit correctors
☐ Horizontal orbit correctors
☐ Interaction point
☐ Marker

SELECT RANGES
Complete octant: none
Straight section: none
Others:

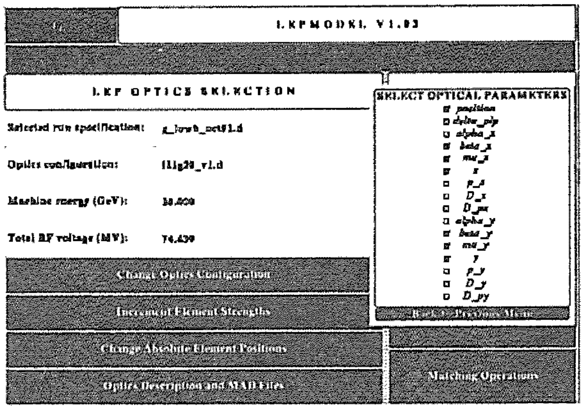
Figure 5: Menu to Select Element Positions in the Machine

- Dialogue allows to define and modify the user interface independently of the application program. The user interface can be built and tried before writing a lot of code.
- Dialogue is quite powerful, and easy to use and maintain.
- Dialogue has been used extensively at CERN.
- Since most of the workstations are Apollos, there is no problem in using a non-portable package.

Dialogue can be used with FORTRAN, Pascal and C programs. FORTRAN requires use of non-standard types like `INTEGER*2`, and of contorted `EQUIVALENCE` constructs to simulate Pascal or C structures. Pascal is little used at CERN. C has been chosen chiefly because it is the main programming language used in the LEP control system.

3.2 Running MAD on a Remote CPU

The LEP Model Program requires a graphics display, thus it must run on the user's workstation. For speed reasons MAD must run on a more powerful computer, like the DN 10000.



Selected run specification: g_lowb_oct91.d

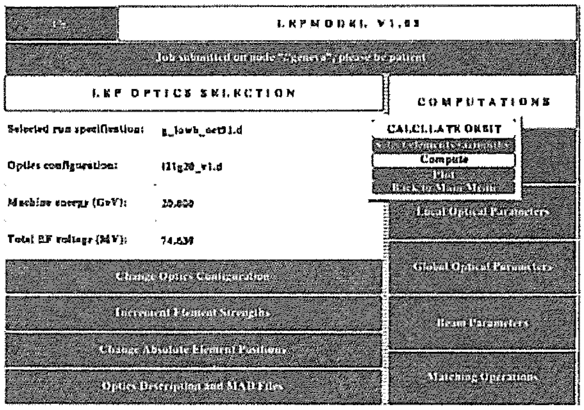
Optics configuration: l11g20_v1.d

Machine energy (GeV): 20.000

Total RF voltage (MV): 74.430

☑ position
☐ delta_p/p
☐ alpha_x
☑ beta_x
☑ mu_x
☑ x
☐ p_x
☐ D_x
☐ D_px
☐ alpha_y
☑ beta_y
☑ mu_y
☑ y
☐ p_y
☐ D_y
☐ D_py

Figure 6: Menu to Select Output Parameters



Selected run specification: g_lowb_oct91.d

Optics configuration: l11g20_v1.d

Machine energy (GeV): 20.000

Total RF voltage (MV): 74.430

Figure 7: The LEP Model Program is Computing...

Unfortunately one cannot use the UNIX standard system call (nor `fork` + `execl`) to spawn another process on th DN 10000. It was found that under the current release of the Apollo UNIX system (SR 10) these calls may crash the operating system. MAD is therefore launched by the Aegis command crp (create process). This command can use the current login name and password of the user, and can be called by the program in a transparent way.

3.3 Data Transfer

The UNIX file system makes it easy to access files from remote nodes. All data transfer uses ASCII files residing in the user's current directory. MAD input files are in ASCII TFS format where feasible, or in MAD input language. MAD output files are written in MAD language, as TFS files, or as listings. This solution allows to develop and test the LEP Model Program independently of MAD by looking at those ASCII files.

3.4 Plotting

Most results come out of MAD as a TFS table. On request the LEP Model Program invokes the LEP dataviewer in a

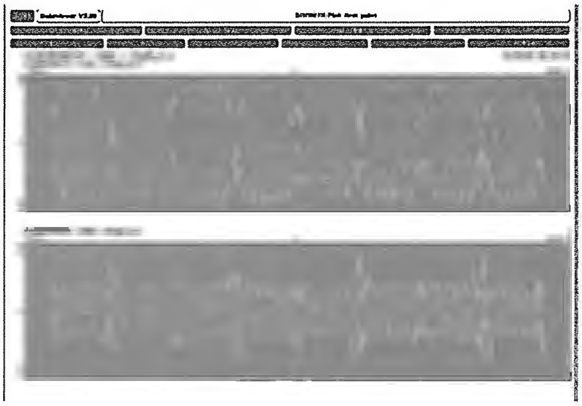


Figure 8: Example of Closed Orbit Plot (one quadrupole displaced)

new window, to plot the newly computed data. The user is free to use dataviewer tools (graph selection, zoom ...) to study the shape of the curves, and a hard-copy of the screen can be made on a Tektronix printer.

4 Future Plans

The LEP Model Program will mainly have to be expanded in the area of interaction with the control system. Operation requires more choices for matching machine parameters, and for introducing the new excitations into the machine. For safety the changes will be introduced by hand, and possibly in small increments; thus effectively creating a "knob" which can be turned to apply a correction to any percentage varying from zero to full value.

In future the program should also generate complete sets of files which can be used to set up a new configuration. Possible options are the following:

- Match excitations to achieve the desired behaviour of the machine, and return the excitation file.
- Compute a table of optical parameters for the machine.
- Launch a program to split the table of optical parameters and to feed the parts into various components of the control system, e. g. the closed orbit correction program.
- Launch a program to set the magnet excitations.
- Launch a program to compute the effect of quadrupole and sextupole strengths onto machine tunes and chromaticities.

5 Conclusions

The LEP Model Program has been found to be a very useful tool for machine setup, as well as for testing new optical configurations during machine development. During operation it has been used successfully for calibration of equipment, to test hypotheses about misalignment and mispowering; this has contributed to the comprehension of various effects in the machine.

The accelerator physicists doing machine development usually have a better knowledge of the MAD program, and they wish to have more freedom in using the LEP Model. A special version of the LEP Model is provided which allows to use private files and/or to edit the files provided by the system. However, this version offers little protection against use of wrong data.

The choice of Domain/Dialogue makes this program modular and expandable. Since the system is event-driven, changes are very simple. To add a new feature it is often sufficient to add a few lines to the interface file, to write and compile a new C routine, and to relink the program without touching any existing C code. Even major rearrangements of the menus are feasible by editing the interface file and by relinking without recompilation.

References

[1] P. G. Innocenti, "The LEP Control System". *International Conference on Accelerator and Large Experimental Physics Control Systems*, Vancouver, Canada, October 30–November 3, 1989.

[2] H. Grote and F. C. Iselin, *The MAD Program, Version 8.4, User's Manual.* CERN/SL/90-13 (AP), Rev. 2, August 1991.

[3] H. Grote and F. C. Iselin, "Recent Developments in MAD Version 8". *International Conference on Accelerator and Large Experimental Physics Control Systems*, Vancouver, Canada, October 30–November 3, 1989.

[4] D. C. Carey and F. C. Iselin, "A Standard Input Language for Particle Beam and Accelerator Computer Programs". *1984 Summer Study on the Design and Utilization of the Superconducting Super Collider*, Snowmass, Colorado, June 23–July 13, 1984.

[5] Ph. Hofmann, *Table File System, Reference Guide.* CERN/LEP LAW Note No. 9 (unpublished).

Optimization of Accelerator Control

N. D. Vasiljev, I. V. Mozin, V. A. Shelekhov
D. V. Efremov Scientific Research Institute of Electrophysical Apparatus
St. Petersburg 189631 USSR

I. INTRODUCTION

Expensive exploitation of charged particle accelerators is inevitably concerned with requirements of effectively obtaining of the best characteristics of accelerated beams for physical experiments.

One of these characteristics is intensity. Increase of intensity is hindered by a number of effects, concerned with the influence of the volume charge field on a particle motion dynamics in accelerator's chamber. However, ultimate intensity, determined by a volume charge, is almost not achieved for the most of the operating accelerators. This fact is caused by losses of particles during injection, at the initial stage of acceleration and during extraction. These losses are caused by deviations the optimal from real characteristics of the accelerating and magnetic system. This is due to a number of circumstances, including technological tolerances on structural elements of systems, influence of measuring and auxilliary equipment and beam consumers' installations, placed in the closed proximity to magnets, and instability in operation of technological systems of accelerator.

Control task consists in compensation of deviations of characteristics of magnetic and electric fields by optimal selection of control actions. As for technical means, automatization of modern accelerators allows to solve optimal control problems in real time. Therefore, the report is devoted to optimal control methods and experimental results.

II. METHODS AND PRINCIPLES OF CONTROL ORGANIZATION

Tasks of the accelerating complex systems control are stated as tasks of extremal control. The following stages may be determined in solving of these tasks:

- study of accelerator as an object of automatic optimization;
- selection of methods of optimization and tracking of extremum;
- comparative study of methods, using models, which have the main peculiarities of the control object;
- synthesis of extremal control algorithm and procedure of estimations of automatic adjustment efficiency at operating accelerator.

Solution of a task can be shown as an example of extremal control of accelerated beam intensity for a proton synchrotron at the Institute of Theoretical and Experimental Physics (Moscow).

Intensity of a beam, injected into the ring, is a function of 11 independent variables, normalized with respect to injector current:

- electrostatic injector voltage;
- injection field intensity;
- radio frequency adjustment in the form of delay of the master clock start;
- correction currents of beam orbit.

Process of intensity change is characterized by spontaneous drift (10 – 12% shift), which can be compensated by varying of the above mentioned variables. Dispersion of an interference is selected in accordance with a noise level, reduced by averaging of beam intensity measurements at the accelerator to 3%.

Criteria of preliminary selection of optimization methods were algorithm discreteness, caused by cyclic processes in the accelerator, as well as convergence in conditions of substantial noises, high speed, minimality of spread in magnitudes of an output value during tuning, compactness of control program.

An important peculiarity, determining selection of a method, is a problem of creation of adequate mathematical description, that forces us to consider an object as a "black box". In this case it is necessary to use search step methods.

It should be noted, that for the use of these methods a necessary condition of object parametrization is satisfied. The condition consists in definiteness of controlled variables, whose varying enables reaching of extremum.

As competitive methods have been selected method of sequential simplex planning, including automatic selection of a step, and methods of random search in modifications:

- with estimation of gradient
- with self-learning
- with punishment of randomness

Values of methods parameters, ensuring a stable convergence and the highest speed in conditions of interferences at models (1) have been determined at the first stage of the studies. In this case a higher speed of the method of sequential simplex planning and higher reliability of extremum search may be noted. One should consider a higher sensitivity in estimation of direction near the extremal zone and complete set of an operating program as the advantages. It is obvious, that it is extremely important to know efficiency characteristics of a priority selected optimization methods, obtained in conditions, near to existing at the object. Comparative studies with the use of models were carried out in the following conditions:

Task of maximization of a single-extremum scalar function

$$I(X) = E_\xi \{Q(X,\xi)\}$$

in situation of noise is considered. Here, $X = (x_1, \cdots x_n)$ is a vector of controlled variables, which are subject to determination. Functional $Q(X,\xi) = I(X) + \xi$ is considered to be measured during optimization. Here ξ is a random value, distributed normally with the expectation, equal to zero, and dispersion σ^2.

Extremum of the function I(X) is determined in the specified region

$$X_i^{min} < X_i < X_i^{max}, \qquad i = 1, 2 \cdots n$$

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S16MS04

Assume, that measurement of a quality function and measurement of vector X are produced only in discrete moments of time

$$t^j = j\Delta t \qquad (j = 0, 1 \cdots\cdots n)$$

magnitudes of vector X of controlled variables on each j-th step are designated as X^j; $I^j = I(X^j)$, $Q^j = I(X^j) + \xi^j$.

A way of determination of a step value $\Delta X^j = X^j - X^{j-1}$ is specified by a search algorithm. Summarization of the methods being compared:

Random search, including punishment of randomness. A next step in the space of controlled variables is determined according to the rule:

$$\Delta X^{j+1} = \begin{cases} \Delta X^j, & \text{if } Q^j > Q^{j-1} \\ \alpha \Xi^{j+1}, & \text{if } Q^j \leq Q^{j-1} \end{cases}$$

where $\Xi = (\xi_1, \cdots\cdots \xi_n)$ is a random vector uniformly distributed on a sphere of unit radius; α is a step scale.

Scale of a step is adapted during search according to the dependence

$$Q = A_0 \exp\{L \times [N^2(1 + \Delta n^2/T^2) + (\sup N)^2(1 - \Delta n^2/T^2)]\}$$

where A_0 is initial scale of a step; N is number of unsuccessful steps, implemented from a last point; supN is maximum number of unsuccessful steps, implemented from any point during all the process of search; Δn is difference between a number of a current point of search and number of a point, where supN of unsuccessful steps has been performed; T is maximum permissible distance between a current point of search and a point, from which sup N of unsuccessful steps has been performed; L is parameter, determining dynamics of step adaptation.

Random search with gradient estimation. Search algorithm has the following view:

$$\Delta X^{j+1} = \begin{cases} \Delta X^j, & \text{if } Q^j > Q^{j-1} \\ \alpha^{j+1} S^j / |S^j|, & \text{else} \end{cases}$$

where α is scale of a operating step; S is stochastic estimate of a gradient, determined by the following algorithm:

$$S^j = 1/(2mg^j)\Sigma_{\psi=1} \{Q(X^j + g^j\Xi^\Psi) - Q(X^j - g^j\Xi^\Psi)\Xi^\Psi$$

where $m \leq n$ is the number of pairs of trials for gradient estimation; g^j is a value of operating step; Ξ^j is vector, uniformly distributed on the sphere of unit radius.

Scales of the operating and trial steps are adapted during search in accordance with the following dependences:

$$a^j = \begin{cases} a^j, & \text{if } Q^j > Q^{j-1} \\ a^j/N+1, & \text{if } Q^j \leq Q^{j-1} \end{cases}$$

$$g^j = \begin{cases} g^j, & \text{if } Q^j > Q^{j-1} \\ g^j/\sqrt[4]{N+1}, & \text{if } Q^j \leq Q^{j-1} \end{cases}$$

N is the number of unsuccessful steps for all the previous process of search. Besides:

a) If the process of gradient estimation reveals positive augments of quality function, transition to a trial point, that is

$$X^j + g^j S^j/|S^j|, \text{ if } \Delta Q^{j+1} = X^{j+1} = X^j + \Delta X^j = (= \max(\Delta Q^{j+1}, \Delta Q^{j,\Psi}))$$

$$X^j + g^j\Xi^\Psi, \text{ else}$$

b) After a step along estimation of a gradient a concluding state is determined by maximum increment of quality function at trial and operating steps, that is

$$X^j = \begin{cases} X^j + g^j\Xi^\Psi, & \text{if } \Delta Q^{j,\Psi} > 0 \\ X^j, & \text{else} \end{cases}$$

Random search with self-learning. Rule of step calculation this algerithm is the following one:

$$\Delta X^{j+1} = (W^{j+1} + R\Xi^{j+1}/|W^{j+1} + R\Xi^{j+1}|$$

where R = const > 0 is radius of guiding sphere; Ξ is vector, uniformly distributed on spere of unit radius; $W^j = (W_1{}^j, \cdots\cdots W_n{}^j)$ is vector of memory, $|W^j| < C$, C = const > 0.

$$W^j = C \operatorname{grad} I(X_0);$$

$$W^{j+1} = lW^j - \delta\Delta Q^j \Delta X^j$$

where $0 < l < 1$ is parameter of forgetting; $\delta \geq 0$ is parameter of self-learning; $\Delta Q^j = Q^j - Q^{j-1}$ is augment of functional at j-th step.

Sequential simplex planning. Essence of optimization by means of this method consists in the following.

Regular simplex, centre of which is at the start point, is constructed in the space of controlled variables, and quality function is estimated at all its vertices:

$$Q(X^j), \quad j = 0,1,2,\cdots\cdots n.$$

Then a trial step – mirror reflection of a worse vertex, where quality function is minimum, is performed through a centre of the opposite face

$$X^{rfl} = 2X^c - X^{wt},$$

where X^{rfl} is reflected vertix position vector; X^c is a vector of the face centre position; X^{wt} is position vector of the worse vertix.

An operating step follows after a trial one to a point, which is determined according to the rule:

$$(1 + \gamma)X^{rfl} - \gamma X^c, \text{ if } Q(X^{rfl}) > Q(X^{wt})$$

is tension;

$$X^{end} = \{\beta X^{wt} + (1-\beta)X^{c}, \text{ if } Q(X^{rf1}) < Q(X^{wt})$$

is compression;

$$X^{rf1}, \text{ if } Q(X^{j}) < Q(X^{rf1}) < Q(X^{wt});\ j \neq wt$$

(If $Q(X^{wt}) < Q(X^{rf1}) < Q(X^{j})$ $j \neq wt$ then X^{wt} is substituted for X^{rf1} and compression is produced). Here, the following designations are taken: X^{end} is position vector of the operating step end; X^{wt} is position vector of the best vertix; $\gamma > 0$ is coefficient of tension; $0 < \beta < 1$ is coefficient of compression.

Subsitution for a vertix, depending on an operating step result, is produced according to the rule;

$$X^{wt} = \begin{cases} X^{end}, \text{ if } Q(X^{end}) > Q(X^{rf1}) \\ X^{rf1}, \text{ if } Q(X^{rf1}) > Q(X^{end}) \\ \quad - \text{ in tension} \\ X^{end}, \text{ if } Q(x^{end}) > Q(X^{wt}) \\ \quad - \text{ in compression} \end{cases}$$

If after compression

$$Q(X^{end}) < Q(X^{wt}),$$

then initial simplex is drawn to the best vertix:

$$X^{j} = 0,5(X^{j} + X^{wt}), \quad j = 0,1, \cdots n$$

Then process is repeated since a moment of determination of a worse vertix.

Study of the methods was carried out with the use of a test nonlinear function in two modifications Q_1 – separable quadratic form, complicated by noise

$$Q_1(X,\xi) = X^{T}BX + B_0 + \xi,$$

where B is diagonal coefficient matrix, determined as approximation of results of statistical identification of an object. B_0 is a free term, ξ is noise addition; Q_2 – unseparable function, having a modular surface of "two-dimensional backbone" $Q_2(X,\xi) = \Sigma B_{ii}X_i^2 = 100(X_{11} - X_{10}^2) = (1 - X_{10}) + B_0 + \xi$

Correctness of statement of a problem on equality of conditions of methods comparison at a specified form of a model consists in identity of initial conditions for all selected methods and optimality of parameters of each method from a viewpoint of a specified characteristic speed of operation, number of quality function samples before reaching of a specified zone of extremum.

All selected methods have been preliminary optimized for parameters. Start points for all tests were a single value

$$X_0 = \{X_i\} \qquad i = 1,2,\cdots, 11$$

Studies were carried out in simulation of optimization process at computer. Results were averaged from ten "ascensions" for the determinate method (simplex) and from fifty "ascensions" for varieties of a random search.

All used criteria of methods comparison are divided into two classes: local and integral ones. The first class is concerned with a single elementary stage of search-operating step, the second one- with all the process of optimization since a start moment till operation of the rule of breakpoint. The following criteria of comparison were used:

- losses for search. An average local rate of optimization is determined and the following calculation is performed:

$$\Pi^{j} = E(K^{j})/E(\delta Q^{j})$$

where

K^{j} is the number of samples at j-th step;
$\delta Q^{j} = \Delta Q^{j}/Q^{j-1}$ is relative change of quality factor;

- error probability. It determines probability of an erroneous operating step $P = P\{Q(X^{j} + \Delta X^{j}) < Q(X^{j})\}$
- inaccuracy. Characteristic of method inaccuracy is integral one. It determines discrepancy ε of the obtained and unknown quatity $E(\varepsilon) = E(X^{*} - X^{ext})$,
  where X^{*} is solution, obtained as result of method operation. X^{ext} is solution, corresponding to extremum of quality function.
- number of samples of quality function for reaching of a specified level. This characteristic depends on initial conditions of X;
- reliability. Reliability $\rho(\varepsilon)$ of method is probability of reaching the specified χ-vicinity of extremum for a specified number of samples of the quality function.

Realiability is numerically estimated in the following way:

$$P(\varepsilon) = 1 - P_{\chi} = \int_0^{\chi} \rho(\varepsilon^{j})d\varepsilon^{j}$$

where $\varepsilon^{j} = |X^{j} - X^{ext}|$ is discrepancy at j-th step, distributed with density $\rho(\varepsilon)$, $P_{\chi} = \int_{\chi} \rho(\varepsilon^{j})d\varepsilon^{j}$ is probability of unachieved required accuracy of solution.

$\rho(\varepsilon)$-noise immunity. One of the most important criteria of the search methods application for solution of the task of accelerator optimization is ability to orient oneself with respect to situation of noise.

Changing level of noise at the accelerator lays down a requirement of stable convergence of the search procedure in some range of noise. Therefore, it is important to study dependence of methods rate on a value of noise and to estimate an upper boundary of a noise level, at which correct orientation is still possible. For collection of noise immunity statistics a noise level is changed from 0 up to 25%, and for each concrete level an average number of steps was determined, necessary for coming to a zone, limited by the surface,

$$\{X: I(X)\} = C \quad C = 0,95\ Imax$$

Main conclusions:

- at the initial stage of search a random search with self-learning is characterized by the smaller losses. At all the following stages the simplex method is characterized by minimum losses. A random search with gradient estimation is similar to it.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S16MS04

- simplex method has the smallest probablity of errors;
- simplex method has demonstrated a higher accuracy of search for extremum in comparison with methods of a random search, has turned out to be the best one in criteria of speed of responce, noise immunity and has demonstrated reliability criterium results, almost identical with the random search method.

III. STUDY OF EFFICIENCY AT AN OBJECT

The studies were carried out for criteria of speed of responce, search variance and reliability. Speed of responce was estimated according to the number of steps, which are to be performed since start till completion of search.

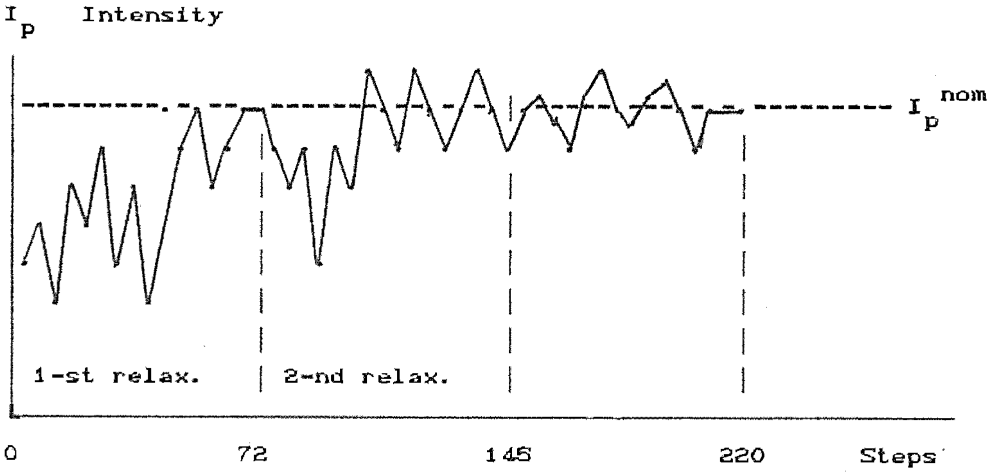


Fig. 1 The extremum search mode of the system

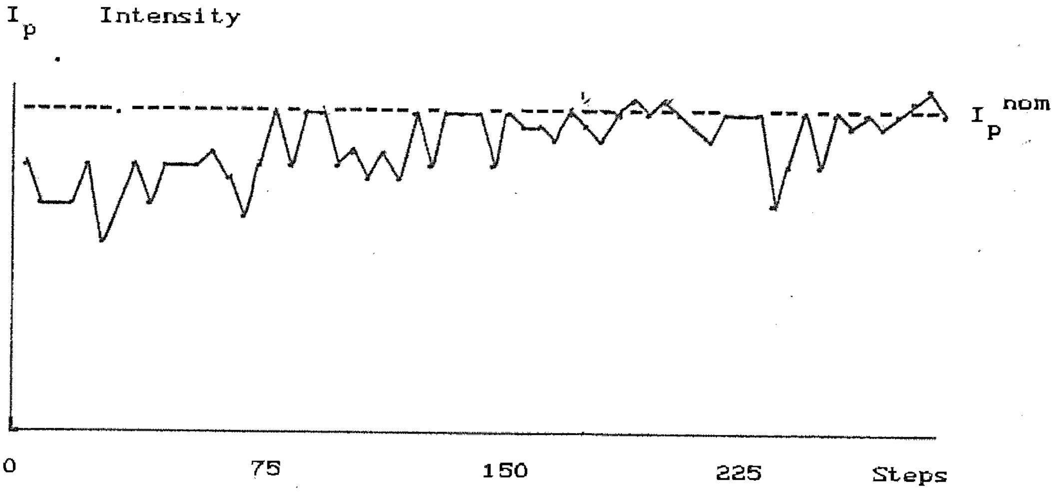


Fig. 2 The extremum follow mode of the system.

Search variance is meant as growth in the output parameter instability because of the search steps. Procedure of extremum search is considered to be reliable, if manual tuning after each series of ascensiens is not effective enough.

Estimation of methods efficiency was carried out in the following conditions:

- detuning in all controlled variables at initial level of intensity of 20 ····· 40% from maximum one;
- detuning of accelerator in two controls (two-dimensional backbone was reproduced artificially);
- nominal conditions of accelerator operation in the presence of the optimal state drift.

All methods turned out to be serviceable ones. However, efficiency of their use turned out to be not identical. In starts from peripheral points the best results were demonstrated by simplex method. An appreciable growth in intensity in the use of this method ends after 60 ····· 80 steps. However, search variance turned out to be high enough. The best average result among methods of random search is 100 ····· 150 steps to an object.

A main result of the carried out studies was proof of applicability of extremal control methods for control of accelerator, and efficiency of them has been demonstrated in practice. This became apparent, first of all, in decrease in time of reaching of the operating conditions of the accelerator (this time was 8 ····· 20 minutes) and in the improvement of the accelerator operation quality. The latter is characterized by process variance, reduced almost by a factor of two, in comparison with manual tuning, and by intensity level, increased by 5 ····· 10%. Fig. 1 and Fig. 2 demonstrate the search and follow modes of system.

IV. CONCLUSION

Though the obtained results are of particular character, they may be used for control of objects of the same class. As for study methodology, it may be assumed as a basis of the approach and organization of solution of the extremal control problems by other types of electrophysical installations.

MODELLING AND OPTIMIZATION OF BEAMS DYNAMICS IN LINAC

N.Edamenko, D.Ovsyannikov, V.Kabanov, A.Zhabko

St.Petersburg State University, St.Petersburg, 198904, USSR
Moscow Radiotechnical Institute, Moscow, 113519, USSR

Abstract

Problems of acceleration and focusing in linear acceleratirs are considered. A general mathematical problem of charged particle beam control is formulated. Methods and algorithms of solving these problems are developed. Problems of mathematical simulation of beam dynamics are discussed in detail. Some beam quality functionals depending on all particle tracks are propossed. Mathematical methods are used for choosing parameters of forming systems. Designed codes allow to simulate and optimize beam dynamics.

This report is devoted to the realization of general approach to problem of dynamical system trajectories control in accelerating and focusing structures.

Let us consider the system of differential equatios

$$\dot{X} = f(t, X, u), \qquad (1)$$

where t is time, X is R^n vectir of phase coordinates, u is R^r control vector and f is vector function. We assume that system (1) has the solution $X = X(t, t_0, X_0)$ with initial conditions $X(t_0, t_0, X_0) = X_0$ for $X_0 \in M_0$, where M_0 is the set of initial values. Let us denote $M_{t,u}$ the shift of set M_0 through trajectories of system (1). Let us suppose that the function $\rho(t,X) \geq 0$ is the system (1) integral invariant and functions $\Phi(t,X,\rho)$ and $G(X,\rho)$ are given and non-negativ.

The main problem is to find the control $u = u(t)$, $t \in [t\ ,T]$, that gives infimum to the functional

$$I = \int_{t_0}^{T} \int_{M_{t,u}} \Phi(t, X_t, \rho(t, X_t))\, dX_t\, dt + \int_{M_{T,u}} G(X_T \rho(T, X_T))\, dX_T. \qquad (2)$$

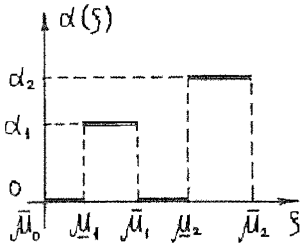
This general approach is used for the charged particles beam control in LINAC [1,2].

Let us consider formulations of cotrol problems related with formating of required accelerating, bunching and focusing regimes for charged particles beams.

The longitudional dynamics of the beam in accelerators with drift tubes may be described by well known equations

$$\frac{d\gamma}{d\xi} = \alpha(\xi)\cos\varphi, \quad \frac{d\varphi}{d\xi} = 2\pi\gamma / \sqrt{\gamma^2 - 1}, \qquad (3)$$

where γ is energy and φ is particle phase, $\xi \in [0,L]$, is longitudional coordinate, L is the length of the structure. In the equation (3) piece-wise constant function $\alpha(\xi)$ is defined by formulas



$$\alpha(\xi) = \begin{cases} 0, & \xi \in [\bar{\mu}_{i-1}, \underline{\mu}_i], \\ \alpha_i, & \xi \in [\underline{\mu}_i, \bar{\mu}_i]; \end{cases}$$

$$i = 1, 2, \ldots, N;$$

$$\mu_0 = 0; \quad \mu_N = L.$$

and proportional to intensity of accelerating field. Let us suppose that energy $\hat{\gamma}$ and phase $\hat{\varphi}$ of particles at the end of acceleratir are given or equal to average particles energy and phase correspondingly. The minimization of functional

$$I = \int_{M_{L,\alpha}} \left[a\left(\frac{\gamma_L}{\hat{\gamma}} - 1\right)^2 + b\left(\varphi_L - \hat{\varphi}\right)^2 \right] d\varphi_L\, d\gamma_L \qquad (4)$$

that characterizes the beam at the end of accelerator, provides optimal parameters.

Let us consider the radial motion now. Let variables η and æ are reduced radial coordinate and velocity of a charged particle correspondingly. Then equetions (3) are copled with system

$$\frac{d\eta}{d\xi} = \text{æ};$$
$$\frac{d\text{æ}}{d\xi} = -\alpha(\xi)\left(\cos\varphi \frac{\gamma\text{æ}}{\gamma^2 - 1} - \frac{\eta \sin\varphi}{\sqrt{\gamma^2 - 1}}\right) - U(\xi)\frac{\eta}{\gamma^2 - 1} \qquad (5)$$

where piece-wise function $U(\xi)$ is intensity of solenoid longitudional magnetic field. To take into account radial dynamics we add integral

$$I_2 = \int_0^L \int_{M_{\xi,\alpha,u}} [c\eta^2 + \alpha æ^2]\, d\gamma_\xi\, d\varphi_\xi\, d\eta_\xi\, dæ_\xi\, d\xi \qquad (6)$$

to functional (4).

This approach gives us opportunity to solve more complicated problems. For example, we can optimize the lattices with beam's space charge taking into account. Let us consider one of such problems. Suppose that particles beam has circle cross-section and almost homogeneous charge density distribution Then the force acting on the particle may be defined by formula

$$F_\eta = \frac{K(\gamma)\eta}{R^2}, \qquad (7)$$

where R is an effective radius of beam cross section. Then the beam's dynamics can be obtained by adding the function F_η to right hand side of second equation system (5).

The choice of the functional (2) allows to define the prelimenary structure of an accelerator and to optimize it to obtain parameters needed.

In monographs [1,2] there is a vast survey of methods of solving different beam's optimal problems. The technique, proposed for solving these problems, allow to construct directed methods of choosing optimal parameters. The analitical formulas for the gradient of control parameters are proposed in these books. In particulary formulas for gradients

$$\frac{\partial I}{\partial \alpha_i}, \quad \frac{\partial I}{\partial \mu_i}, \quad \frac{\partial I}{\partial \bar{\mu}_i}, \quad i = 1, 2, \ldots, N$$

for above mentioned examples are developed.

We produced codes for IBM PC compatible computers for solving these problems. One of these codes provides optimal parameters for solenoids, quadrupols and gaps with drift tubes. Authors wanted to make the program allowing in interactive regime

- -to simulate charge particles beam's dynamics;
- -to formulate conditions for the beam configuration in the space of coordinates and speeds at the end of structure;
- -to calculate feasible structure parameters to satisfy formulated conditions.

Main assumptions are

a) the particle interaction isn't taken into consideration;

b) electromagnetic field amplitudes are piece-wise constant.

Under this hypothesis equations of beam's dynamics allow the analytical solution everywhere exept accelerating gaps. The aim is to place particles into domain, bounded by curve on phase plane. Let vector Y is (W, φ) or (τ, τ') or (x, x') or (y, y') and $S(Y) = C$ is equation for boundary. Let the function $F(Y)$ is defined by formulas

$$F(Y) = 0, \quad \text{if} \quad S(Y) < C$$
$$\text{else} \quad F(Y) = (S(Y) - C)^2.$$

Let the minimizating functional is the integral of function $F(Y)$ by the particle set. We shall minimize it by varying elements lengths and values of electric and magnetic fields amplitudes. The functional's gradient can be obtained with the help of conjugate system that is solved analitically. The restrictions on control are considered during optimization The phase variables $(t, x, y, \dot{x}, \dot{y}, \dot{z})$ values at the end of every structure element are computed and saved. These data is used for construction of beam's dynamics vizualization that is made with the help of graphics and tables. The user can enter the accelerating and focusing structure, particle's type and initial beam phase configuration. The program made it possible to watch the 2D & 6D beam dynamics and to optimize the structure's parameters. If the user want to optimize the structure he should set the beam phase configuration at the end of the structure and put restrictions of the control. User can control the process of optimization. The codes can be installed on IBM PC/AT 286.

The initial and optimized beams cross-section are depicted on figures 1,2,3,4.

The beam's projections on the plane (W, φ) at the end of initial and optimized structures are depictured on fig.1 & 2. The ellipse bounds the set desired.

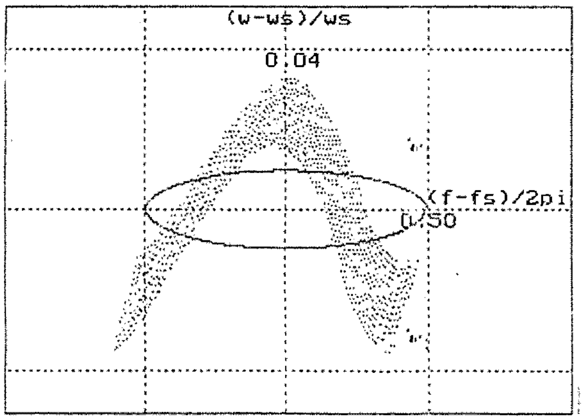


Fig. 1

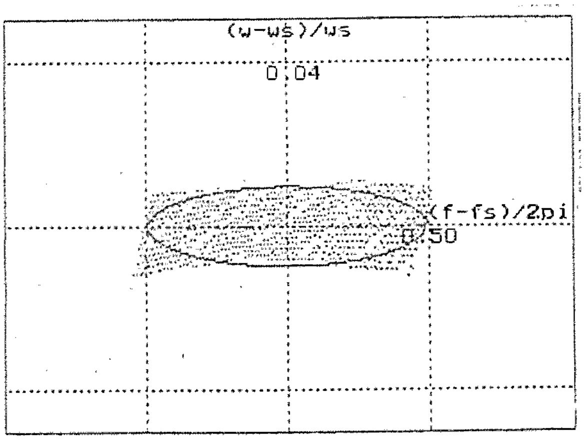


Fig. 2

The beam's projections on the plane (τ,τ') at the end of initial and optimized structures are depictured on fig.3 & 4 correspondingly. The ellipse bounds the set desired.

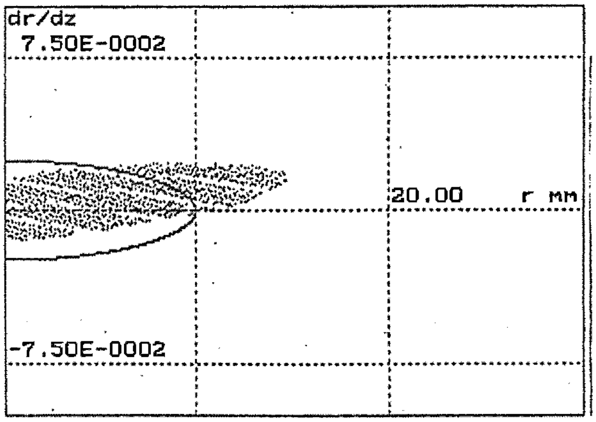


Fig. 3

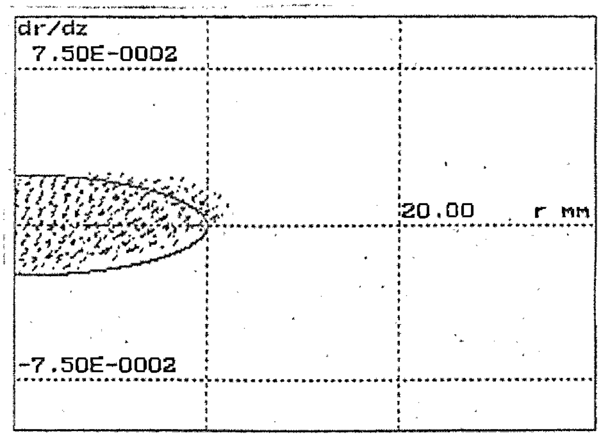


Fig. 4

References

1. Ovsyannikov D.A. Mathematical methods of beams control. L. : Leningrad University, 1980 (in Russian).
2. Ovsyannikov D.A. Simulation and optimization of charged particle beam dynamics.L.: Leningrad University, 1990 (in Russian).

Development of a Diagnostic System for Klystron Modulators Using a Neural Network

M.Mutoh, T.Oonuma and Y.Shibasaki
Laboratory of Nuclear science, Tohoku University
1-2-1 Mikamine, Taihaku-ku, Sendai-shi, 982, Japan

I.Abe and K.Nakahara
KEK, National Laboratory for High Energy Physics
1-1 Oho, Tukuba-shi, Ibaraki-ken, 305, Japan

Abstract

The diagnostic system for klystron modulators using a neural network has been developed. Large changes in the voltage and current of the main circuit in a klystron modulator were observed just several ten milli-seconds before the modulator experienced trouble. These changes formed a peculiar pattern that depended on the parts with problems. Diagnosis was possible by means of pattern recognition. The recognition test of patterns using a neural network has shown good results. This system, which is built in a linac control system, is presently being operated so as to collect new trouble patterns and to carry out tests for practical use.

I. INTRODUCTION

In the electron linac, high-power klystrons are used as an amplifier that provides rf power to accelerate electron beams. Five modulators driving five klystrons are installed at Tohoku University's 300 MeV electron linac.

Since a klystron modulator, which generates pulsed power output of high voltage and a large current, is operated under severe conditions, it has problems most frequently among the devices in a linac. The Tohoku linac is at fist adjusted by the accelerator group and is then operated under regular conditions by experimentalists who use the linac for their experiments. However, they are not always specialists in the accelerator field. When the various devices comprising a linac have problems, it is necessary to install support systems for linac operation in order to suitably dispose of these problems and to continue linac operation. Therefore an expert system for the diagnosis of beam operation [1] and the diagnostic system for a klystron modulator have been developed.

In designing this system, it was noticed that large changes in the voltage and current of the main circuit in a klystron modulator existed just several ten milli-seconds before the modulator had a problem. These changes formed a peculiar pattern that depended on the parts with problems. Some interesting patterns were observed in preliminary tests [2]. Diagnosis was possible by means of pattern recognition. A neural network having an excellent ability for pattern recognition was used for comparisons between learned and actual patterns. It was useful to apply the neural network to this system in order to improve the accuracy of the diagnosis, to simplify diagnostic programs and to reduce the development period. In order to increase the accuracy of this system, more trouble patterns should be learned; as of now, very few patterns

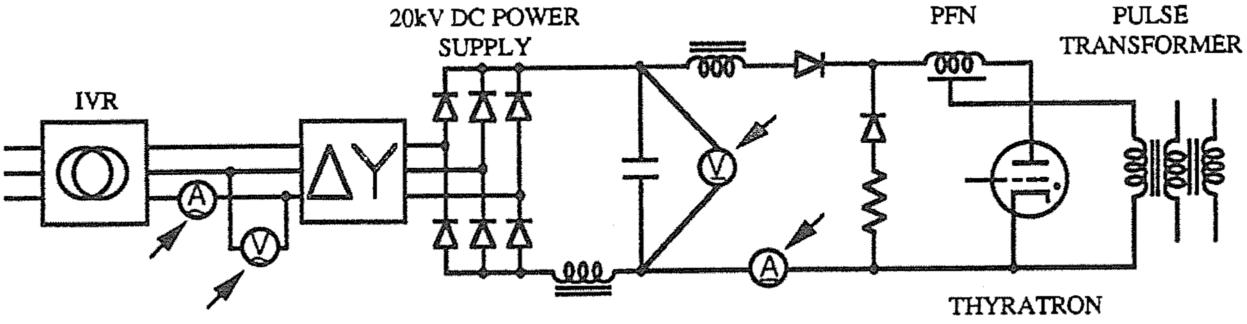


Fig.1. The klystron modulator circuit.

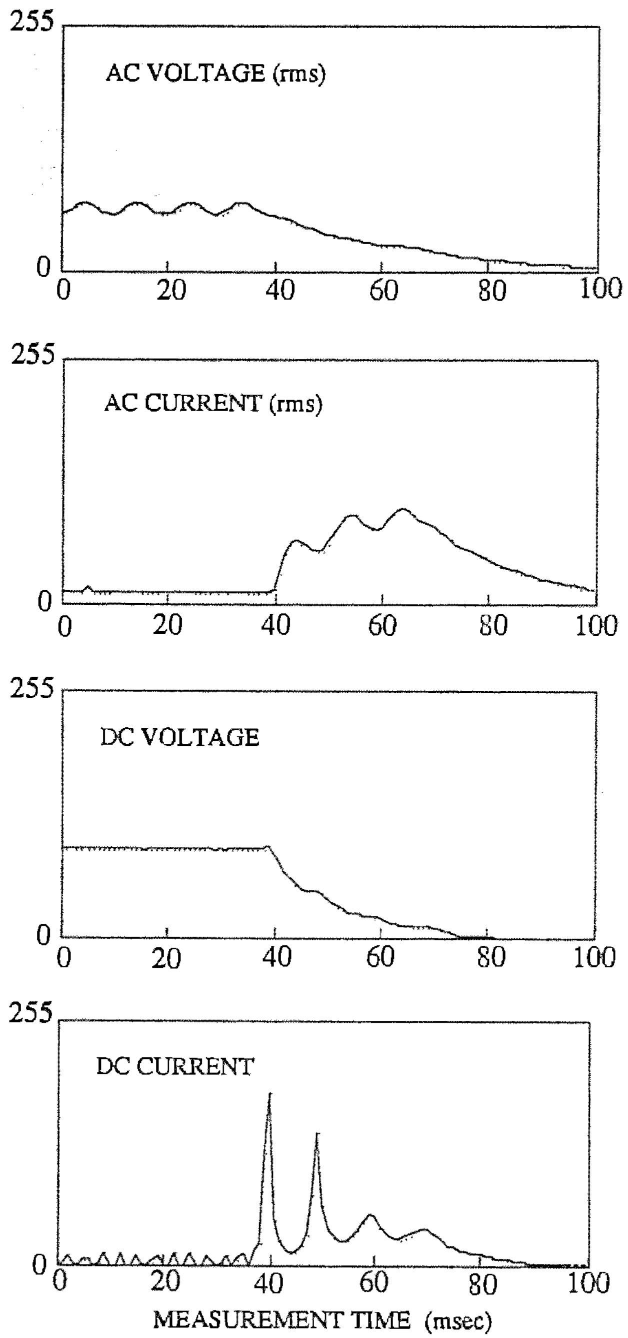


Fig.2. The typical pattern observed just before shutdown. Large current was flowed by a continuous conductive state in the thyratron and large voltage drop was simultaneously caused. Y-axis is an output value of ADC. Each curve is constructed by 100 sampling data.

have been collected. This system has been constructed and operated so as to collect new trouble patterns and the system diagnoses two klystron modulators for practical use.

This system comprises data-taking equipment built into each klystron modulator, as well as a personal computer with a simulation function of the neural network. The personal computer is also associated with the linac control system. Any diagnostic result is communicated to a linac control computer, and is shown on a display of the operater's console in the control room.

II. CONSTRUCTION

A. Klystron modulator

Fig.1 shows the main circuit in a klystron modulator. A pulsed high voltage and a large current are generated by composing of a pulse-forming network (PFN) and a thyratron, stepped up by a pulse transformer and supplied to a klystron. These signals, which are used for diagnosis of the modulator, are measured at four positions, which are shown arrow mark (fig.1); these are the AC voltage and its current, as well as DC voltage and its current. Fig.2 shows a typical pattern observed just before shutdown; this pattern shows the situation of an over current caused by a continuous conductive state in the thyratron. These signals are continuously measured by a data-taking equipment, as shown in fig.3. The data-taking equipment comprise isolation amplifiers, special elements to convert AC rms to DC (rms/DC), a multiplexer (MPX), a sample and hold (S/H), an 8-bit analog to digital converter (ADC), a memory to store the measured pattern data, an asynchronous serial communication interface adapter (ACIA) and a microprocessor (Intel 8085) used to control all of this equipment. These signals are sampled regularly at intervals of 1 msec and the data of 100 samples are stored in memory. When this equipment receives a shutdown signal from the klystron modulator, it immediately stops sampling these signals and sends the pattern data (4 positions x 100 samples) to a personal computer installed in the control room with ACIA (9.6 kbit/s) through an optical fiber cable.

B. Neural network

The personal computer FMR-70HX3 (Intel 80386, 25MHz, MS-DOS or OS/2) manufactured by Fujitsu is used. It has the function of a neural network simulator. This simulator works by combining simulation software (NEUROSIM/L) and an exclusive board (neuro-board) with a digital signal processor (MB86332) for high speed floating-point operations and 4 M bytes of memory for interconnections between neurons. The simulator is capable of making up to 1000 neurons at most, and the process speed is 4 M connections/s at most. The NEUROSIM/L functions as a simulator that can be accessed by user's programs with C language, and a tool for both learning and recognizing through MS-WINDOWS. The neural network in this system has a 3-layered structure; the number of the neurons in each layer is 60 in an input-layer, 10 in a midle-layer

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S17AIA01

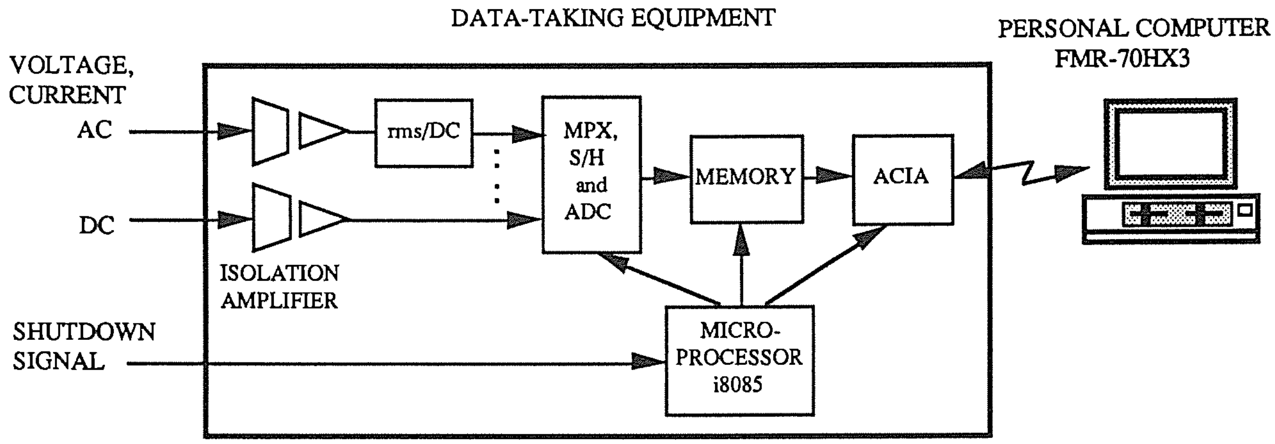


Fig.3. Block diagram of the data-taking equipment.

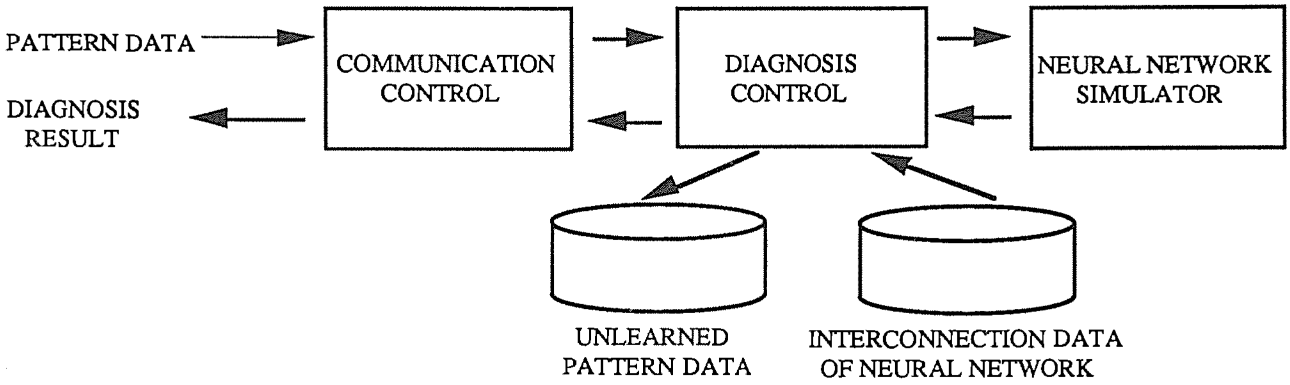


Fig.4. Software architecture in the personal computer.

and 5 in an output-layer, respectively. The abnormal part of the pattern accepted from the data-taking equipment is added to the input-layer. Each output in the neural network simply corresponds to each cause of trouble. A back propagation method was used for learning. The learning time depends on the learning parameters; the optimum value of the parameters are obtained by repeating the learning process trials. It then takes a few hours for this process. Afterwards, the constructed neural network is stored in the disk of the personal computer and is then loaded into the memory in the neuro-board when this system starts. Although the learning process has been used as only an exclusive tool, the new version of NEUROSIM/L has been so improved as to be able to set various learning parameters and to be operated the learning by the user program without any manual operation. The system is therefor able to provide an automatic learning function.

The program used for diagnosis in the personal computer comprises a simulation unit of the neural network, a diagnosis control unit and a communication control unit (fig.4). Before the diagnosis, the interconnection data of the neural network which has already been learned are loaded from the disk into the memory of the neuro-board. The pattern data for the diagnosis sent from the data-taking equipment is processed so as to supply the input-layer in the neural network at the diagnosis control unit; it is then sent to the simulation unit. The diagnostic results obtained at the simulation unit are returned to the diagnosis control unit. If the diagnosed pattern is applicable to the pattern already learned, the diagnostic results are sent to linac control computer so as to inform the linac operator; if not, the pattern data from the data-taking equipment are stored in the disk for the next learning process.

C. Diagnostic system in the linac control

Fig.5 shows the diagnostic system in the linac control system. In the linac control system [3], an original control loop of a Micro VAX-II is connected with magnet power supplies, the klystron modulators and graphic displays on the console

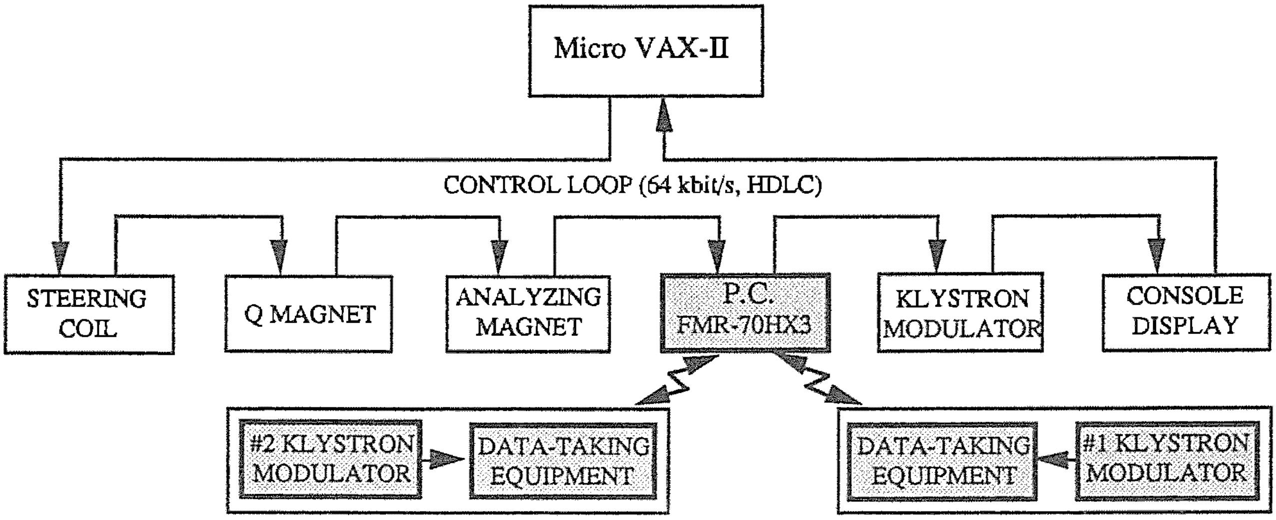


Fig.5. Block diagram of the linac control system, including the diagnostic system for the klystron modulator.

desk. The control loop is a 64 kbit/s synchronous transmission mode based on the High-level Data Link Control procedure (HDLC). Intelligent controllers based on a VME bus, which control the devices comprising the linac, are connected to each node of the control loop. The diagnostic results from the personal computer are sent to the Micro VAX-II through the control loop; they are also informed to the linac operator by graphic display and recorded on the linac operation log, together with other useful information from the linac.

III. CONCLUSION

As of now, various tests have been conducted. In the future, increasing the amount of pattern data will become a serious problem, such as increasing the learning time and establishing how to learn efficiently. In order to increase the learning time, the extraction of special features from a pattern and reducing the total number of neurons in the neural network must be improved. As for the learning method, this system should provide an automatic learning function, so that it can work when it obtains a new pattern, relearns without any manual operation and changes an old neural network to a new one. The addition of an automatic relearning function to this system is in progress.

As our next step, this diagnostic system will be associated with an expert system for the diagnosis of a klystron modulator [4] in order to realize higher accuracy.

IV. REFERENCES

[1] Y.Shibasaki, M.Mutoh and I.Abe, "Expert system for the diagnosis of beam operation trouble," Proceedings of the 16th Linear Accelerator Meeting in Japan, Tokyo, September 1991,pp.333-335.

[2] M.Mutoh et al., "The diagnostic system of klystorn modulator using neural network," Proceedings of the 15th Linear Accelerator Meeting in Japan, Sapporo, September 1990, pp.251-253.

[3] M.Mutoh et al., "Improvement of the control system for linac and pulse beam stretcher at Tohoku university," Proceedings of the 7th Symposium on Accelerator Science and Technology, Osaka, December 1989, pp.240-242.

[4] I.Abe et al., "Expert system for diagnosis of klystron modulator," LINAC conference 1990 in Los Alamos USA, September 1990.

Diagnostic Expert System in the PF LINAC

Isamu Abe, *Masaharu Kitamura and Kazuo Nakahara

Photon Factory, National Laboratory for High Energy Physics (KEK)
1-1, Oho, Tsukuba-shi, Ibaraki-ken 305, Japan
*Department of Nuclear Engineering, Tohoku university
Aramaki-Aza-Aoba, Sendai, Miyagi 980

Abstract

A prototype diagnostic expert system (ES) was developed for the Photon Factory 2.5-GeV electron / positron LINAC injector system. The ES has been on-lined with the conventional linac computer network for receiving real data. This project was undertaken in an attempt to reduce the linac operator's mental workload, diagnosis duties, and to explore Artificial Intelligence (AI) technologies.

The outlook for ES and its problems, and what has been achieved are outlined in this presentation.

I. Introduction

Diagnostic problems are relatively well understood both empirically and theoretically. A variety of shells / tools are available on the market to facilitate the implementation of diagnostic systems. We have developed several diagnostic ES for the LINAC and some are now under operation. Having gained experience through previous projects, we built a new hybrid ES this time. The application described here is an ES for the injector system of the Photon Factory (PF) LINAC[1-4], which is being operated a total of 5000 hours per year, making injections to the PF storage ring and the TRISTAN e+/e- collider. Accelerators (LINAC) are complex devices, using many thousands of components. We have been looking for appropriate expert system shells and tools with which we can easily and rapidly establish an expert system. For several years, a small ES based on a personal computer was used for exploring applications of AI techniques. A prototype diagnostic system has been built in order to determine whether or not the various problems can generally be solved using an ES frame-work; a knowledge base (K/B) for the accelerator domain and task analysis were also investigated in this project.

II. Why we need ES for the accelerator

When any fault or trouble occurs in the LINAC, the operator is required to recover the system, even at midnight, even though, he may not be an expert regarding many of the fields required to diagnose the specific trouble.

Diagnosing faults in a complex process is a task that requires experience and considerable knowledge in many fields. Thus, any assistance given to the linac operator regarding diagnosis and operation is extremely desirable.

When human experts are scarce, and when problems must be solved for which there are no established solutions or exact theories for problem solving, an expert system seems to be appropriate. When there are several candidate procedures, or algorithms, involved in problem solving. Also, ES should be useful and more efficient than conventional programming.

Since most ES are flexible, if we change the physical structure of the accelerator, the K/B can be gradually extended by adding new knowledge while being refined. Programming costs will be minimized by using ES. This is the essential advantage of an expert system.

III. Definitions of AI

Some people think that "AI by itself can solve all of the problems." We must be very careful concerning this idea. AI is not magic, and its capability is still limited in solving practical problems . On the other hand, there are other people who believe that AI can do nothing worth while at all. We, thus, need to define AI before any discussion.

We have seen many definitions. People's dreams are big, and they could have answered that "Artificial Intelligence is the science of constructing a thinking machine."

Marvin Minsky gave a new definition: "Artificial Intelligence is the science of making machines do things that would require intelligence if done by men".

Today, the abbreviation "AI" is used with the meaning of "Advanced Information processing technology". From this perspective, AI will certainly become more and more important in the Accelerator domain.

We discuss here only knowledge-based systems which are a subset of AI technology. In most cases, ES is a rule-based production system which dispenses with specialized knowledge of a well-defined domain.

It is said that ES belongs to the most important developments of a new type of software generation.

IV. History of AI

It should be mentioned that AI is still very young. Fundamental AI researches are necessary for continuously defining and realizing AI's future.

The term "Artificial Intelligence" was invented at the Dartmouth Conference in 1956, where John McCathy (Stanford) and Marvin Minsky (from MIT) were participants. After that, a new field of research was born.

The feasibility of the first expert system was demonstrated in the 1970's under the leadership of Edward Feigenbaum. There have been many successful ES in the past. If we classify the generations[6], from the viewpoint of an ES tool, we can define three generations as follows:

In the 1st generation, until 1980, simple tools and languages were utilized;

In the 2nd generation, 1980 to 1985, hybrid tools were commonly used.

We are now in the 3rd generation, which is characterized by easy-to-use tools and task oriented; domain-specified tools are available now. We have established commercial systems, specialized languages, and tools for developing such systems.

Today, the development of the so-called knowledge based systems has started moving out of the research laboratories of pure informatics into other disciplines of science.

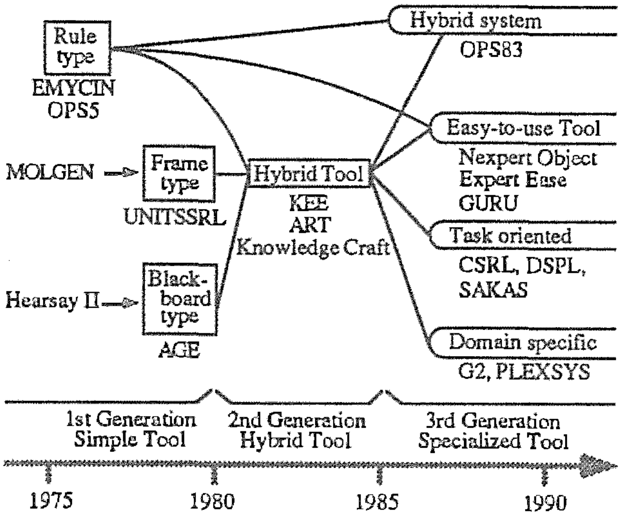


Fig. 1 History of Tool

V. Software tools

For the most part, the ES tool comprise an inference engine, knowledge-aquisition, a knowledge-base, a explanation system and a user interface.

An ES tool should provide at least the following features:

- transparency and portability of the knowledge base,
- expert-like diagnosis (less questions, learning,)
- a graphical user interface for representation and a function of explanation

are very important in knowledge handling. Several operation modes are also necessary, such as a Run mode, Test modes and interactive debug modes.

Some useful knowledge-aquisition tools are available on the market which are helpful for simple surface knowledge.

VI. Hardware/software configuration

The LINAC injector control system has several microprocessor units (MPUs) which are connected to a local network (LOOP-2).

When a fault occurs in the injector part, the MPU picks up the first reason for the trip down and sends a message to a minicomputer, MELCOM 70, in the subcontrol room through LOOP-3, LOOP-2, and CAMAC. There are 50 interlocks and analog data displaying at the local pulser panel. Of course, ES requires more information for a complete diagnosis. Most of the same data which is available to checks locally is transfered to the main computer and as well as to the ES. The ES would be triggered to start diagnosis upon receiving a fault message through the network.

The ES has the following configuration: HP9000/370 (development station), 375 (LINAC operator's console), HP-UX, C, NEXPERT-OBJECT (hybrid-tool), Data-View (graphic tool), equipped with a 16MB RAM, 300MB(600MB) hard disk, and 3.5" floppy disks. There is a pre-process station (FMR-50 personal computer) which has a 600MB magnetic optical compact disk (CD), and is connected to the DSLink. There are about 30 personal computers (PC) under the DSLink network, and each PC has its own purposes or functions such as gateway, server, accelerator operator's console, monitors, development, expert system, and OS/2 stations.

The DSLink operator's console network of the LINAC is connected to the conventional LINAC networks through the gateway (FMR-70 HX3).

In addition to rule-like knowledge, waveform information is also highly important in carrying out diagnostic tasks. Pattern recognition using a neural network is thought to be helpful in this context.[6] A neural network is well suited for pattern-recognition problems, but has a disadvantage that the learned knowledge is hidden in the weights of the network's connections. We have thus developed what is called a hybrid ES by combining a neural network.

The injector ES has a debugging mode for diagnosis which makes it easy for users to carry out simulation at any time. The ES can operate in two modes (AUTO or MANUAL) at the operator's discretion. In the AUTO mode, when the ES receives serial shutdown data from a pre/post process station through the network, it automatically and periodically starts the diagnosis procedure. During a periodic analysis, ES starts a monitor programme to collect raw data for the error diagnosis process during the data-taking phase. The interactive debug mode can be invoked if an error in the run or one of the test modes occurs. Symptoms will be derived and possible repair actions will be proposed. In the debugging mode, the inference action is initiated, or fired, manually by the knowledge engineer or user. Each rule must be simulated step by step in this MANUAL mode both before and after running using empirical knowledge.

VII. Defining the task and application domain

In the accelerator domain, ES would be useful for operation support, fault diagnosis, and design work. We have investigated the task of diagnostic ES for the injector of the LINAC. The following tasks were important for ES:

1) On-line data processing.
2) Finding heuristic methods of diagnosis from human expert (surface knowledge).
3) Local checks using oscilloscopes, measurement tools, and visual checks.
4) Logbook, chart, and drawing checks.
5) Modeling, and calculations (deep knowledge).

These are the fields that an expert carries out during diagnosis; an expert system should thus duplicate them.

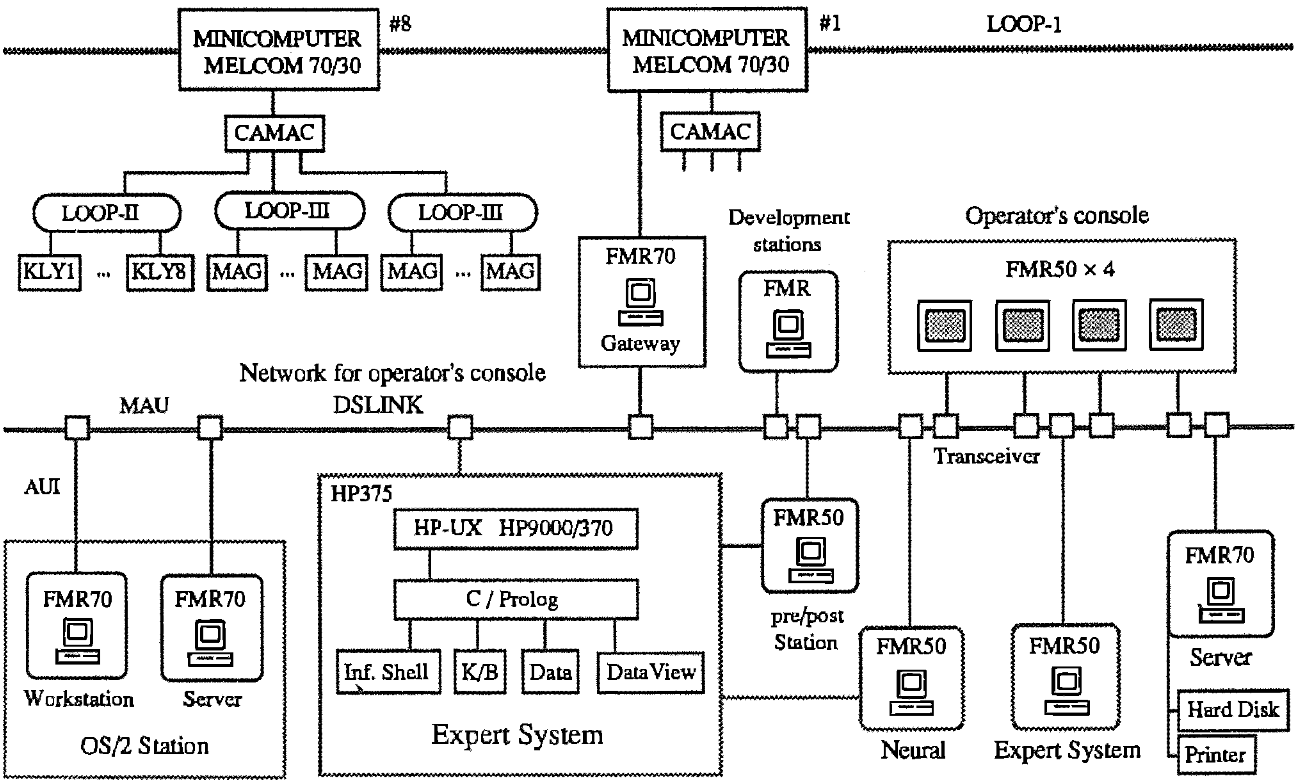


Fig. 2 Expert System
and the PF Linac Control console network

VIII. Diagnostic ES and the Problems

The most difficult problems to be resolved have been the acquisition and representation of knowledge, as well as the selection of an inference strategy. When we develop an AI system, we must be clear about the following items regarding the planned applications:

1) The diagnostic resolution must be determined .
2) Necessary information must be available on-line; otherwise,the operator must supply a large amount of data to the expert system, which is quite unrealistic in practical situations.
3) The introduction of deep knowledge, such as model based diagnosis, is desirable.
Without deep knowledge, the number of required rules can be quite large.
Domain-specific knowledge can be represented in a logical programing system, in a frame-based system, in rule-based systems, or in hybrid systems.
4) The strategy for controlling the inference procedure is critically important.
This strategy should be made consistent with the content of rules in the knowledge base
5) Inference mechanisms: We must determine which direction of reasoning is appropriate, a forward-directed strategy from an initial to a goal state, or a backward. Inference method may be deductive,abductive, or inductive.
6) The handling of uncertainty in knowledge as well as in the observed data is also necessary.
7) Appropriate actions should be implemented.
If a new unknown problem occurs, the system should propose to contact a field expert.
The system must assist the expert in updating the related part of a knowledge.
8) Is a learning system or Case-Based Reasoning (CBR) important ?

IX. Knowledge Acquisition

Generally, knowledge acquisition can not be based on any model in this kind of diagnosis. Knowledge engineers or experts must build knowledge-base extracting rules and procedures using an empirical approach through interviewing experienced field experts regarding both operation and repairing.
Here, we have a bottleneck concerning knowledge taking.

For making the LINAC injector diagnostic knowledge-base, about one month during the first stage has been spent in interviewing and knowledge engineering for a shallow target. As the second stage, we are refining the knowledge base in order to obtain more knowledge. One hundred expressions concerning production rules, some frames, and object are being handled in the ES.

X. Conclusions

The diagnostic capability is still limited; moreover, we have started with a shallow target. It was sufficiently faster than human (operators) diagnosis regarding execution.
A new K/B(Knowledge Base) is now being added and tested for expected phenomena in order to obtain more reliability.

It has become feasible now in the Accelerators field. We are sure that expert system technology will provide new possibilities for software systems, especially in future accelerators. Expert systems should be used more in our field, as in others.

As a final statement, the following will be the areas in which we must carry out research:

How to deal with a large domain,(many tasks).
How to model applications, such as task and domain analysis.
We need portability of K/B.
Learning systems and Case Based Reasoning.

Acknowledgements

The author would like to thank Dr.S.Ohsawa, M.Yokota and A.Shirakawa for their help in the present phase of knowledge acquisition, as well as Professor A.Asami, Director of the Injector-LINAC division in KEK, and Prof. K.Sugiyama of Tohoku University for their support to this project.

References

[1] I.Abe, H.Hanaki, S.Anami and K.Nakahara "Klystron Modulator Trouble Shooting" LINAC conference (1989)
[2] A.Shirakawa, I.Abe, and K.Nakahara "OS/2 AND DSLink FOR THE PF LINAC CONTROL" 16th LINAC conference (1991)
[3] I.Abe, M.Fujieda, "DISPLAY SYSTEM FOR CONTROL CONSOLE OF KEK Proceeding of the 6th symposium on Accelerator Science and Technology (1987)
[4] K. Nakahara, I. Abe, R.P. Bissonnette, et,al Nucl. Instr. and Meth.A251(1986) 327-336
[5] K.Nakahara, I. Abe, A.Enomoto, Y.Otake, and T.Urano Nucl. Instr. and Meth.A247(1986) 153-158
[6] AI Hakusho ICOT-JIPDEC AI center
[7] Isamu ABE, Masashi Fujieda Proceeding of the 6th symposium on Accelerator Science and Technology(1987)
[8] M.Mutoh, T.Oonuma, Y.Shibasaki, Isamu ABE and K. Nakahara "Development of a Diagnostic system for Klystron Modulators using a Neural Network", This conference

GLAD: A GENERIC LATTICE DEBUGGER*

Martin J. Lee
Stanford Linear Accelerator Center, Stanford University, CA 94309

Abstract

Today, numerous simulation and analysis codes exist for the design, commission, and operation of accelerator beam lines. There is a need to develop a common user interface and database link to run these codes interactively. This paper will describe a proposed system, GLAD (Generic LAttice Debugger), to fulfill this need. Specifically, GLAD can be used to find errors in beam lines during commissioning, control beam parameters during operation, and design beam line optics and error correction systems for the next generation of linear accelerators and storage rings.

INTRODUCTION

I have been asked to present a paper on Model-Based methods and Artificial Intelligent (AI) methods for accelerator control. Being a teacher of T'ai Chi, a system of Chinese exercises based on the Yin and Yan principle, I am familiar with the Taoist saying:

Tao is Yin and Yan

It is natural for me to think of Model-Based methods and AI methods as one system of methods—the GLAD system. While thinking about the GLAD system, I made a list of the Yin and Yan pairs associated with accelerator control:

Yin	Yan
Beam line	Beam
Design	Control
High-level	Low-level
Commission	Operation
Play back	Real time
Beam	Parameter
Element	Strength
Look	Adjust
Off-line	On-line
Inverse-Modeling	Modeling
Interpretation	Analysis
Automatic	Manual
Solution	Problem
Rule-based	Trial-and-error
Prediction	Validation
Future	Present
Waste Prevention	Risk Reduction

The purpose of this paper is two-fold: (1) to describe the Model-Based control philosophy in terms of these Yin and Yan pairs, and (2) to propose the GLAD method as a practical way to upgrade any existing accelerator control system to become an intelligent Model-Based control system.

THE TAO OF MODEL-BASED CONTROL

The Tao of Model-Based Control is a generic way of controlling beam parameters using modeling and simulation codes interactively during commissioning and operation of a beam line.

Beam Line Design and Beam Control Codes

Every accelerator or storage ring system consists of a charged particle beam propagating through a beam line composed of bending, focussing, and accelerating elements. In the design stage, the effects due to errors in the beam line are simulated using modeling codes. For example, modeling codes are used to design an orbit correction system consisting of dipole correctors and beam position monitors (BPMs). During commissioning and operation, these same modeling codes can be used to find the errors in the beam line elements and to control beam parameters interactively.

High-level and Low-level Software

The software of a Model-Based control program can be divided into high-level and low-level software. High-level software is the modeling and simulation code for the design and control of an accelerator beam line. Low-level software is the application code for setting the strengths of the beam line elements and measuring the beam parameters.

Commissioning and Operation Goals

High-level software can be subdivided into two types: one for commissioning and the other for operation. The goals of commissioning and operation are not the same. The goal of commissioning is to find the causes of measured beam errors, while the goal of operation is to correct the error effects on the beam.

Here is one example. Often beam orbit errors are caused by magnet misalignments and BPM reading errors. During commissioning, it is necessary to first use orbit simulation codes to find errors in the beam line elements (the sources). After these errors are found in the beam line elements, they can be incorporated directly into the "as-built" model. During operation, the same orbit simulation codes can be used to identify the best correctors and calculate the strengths needed to correct the errors. Since the success of the operation will depend on the accuracy of the as-built model, the primary objective of commissioning is to find an accurate model of the as-built beam line.[1]

Play-back and Real-time Applications

In general, the procedures to find the "as-built" model involve the following two-step procedure:

1. Measure specific beam parameters, and
2. Analyze the measured data.

---

* Work supported by Department of Energy contract DE-AC03-76SF00515.

The first step uses low-level real-time software and the second step uses high-level play-back software. The database of the control system provides an interface between high and low-level software.

A GENERIC LATTICE DEBUGGER (GLAD)

Today, numerous accelerator simulation and analysis codes exist that can be used to find the as-built model of a given beam line. To name a few, there is COMFORT, [2], DIMAD [3], MAD [4], PETROS [5], RESOLVE [6], TRACY [7] and TRANSPORT [8]. In general, the GLAD process will find the as-built model using the following procedure:

1. Measure orbits, tunes, profiles, etc.
2. Save the measurements in the database.
3. Translate data files to the input format of a particular code,
4. Analyze the measured data to find errors in the beam line or in the model.
5. Validate errors with beam tests.
6. Update the model.

In addition, if errors exist in the quadrupole strengths, the beam line can be "tuned" using the as-built model. The tuning process typically involves:

1. A calculation of the "lattice function" errors (such as mis-match in the beta-functions, eta-functions, etc.), and
2. An adjustment of the lattice functions to remove the error.

If quadrupole magnets are misaligned, they can be corrected using orbit correctors near the misaligned elements. This correction process typically involves the following steps:

1. Measure the orbit.
2. Identify and correct BPM offset errors.
3. Calculate the strength of the correctors using an orbit simulation code such as RESOLVE.

The GLAD process will be either an "off-line" process or an "on -line" process. When file translation (step 3 of the GLAD process described above) are done manually, it is considered an off-line process. When they are done automatically, it is considered an on-line process. The advantage of an on-line process is to reduce the turn-around time so that experimental validation can follow error prediction as quickly as possible.

The GLAD system can be used either manually now or automatically in the future. For manual applications of the GLAD system, the graphic interface allows a user to implement these procedures interactively. With "adaptor" codes to link the user interface to the database of the control system, the Model-Based commissioning and operation procedures can be implemented on-line directly. An example of manual application is to use RESOLVE to validate the model and COMFORT to tune the lattice. In the future, rule-based expert systems can be added to the GLAD system to perform these tasks automatically.

A Generic Interface [9] (GENI-X) is being developed to run modeling or simulation codes and to display the input and output data using X-Windows. To link a new module in the GLAD system to a given database requires two adaptor codes: DB-Get and DB-Put. The DB-Get adaptor code is used to translate files from the database format to the input format of the modeling code. The DB-Put adaptor is used to "download" the results into the database of the control system. In this way, the GLAD system can link up with any existing control system. In addition, GLAD can also be used for developing rules and procedures to find errors automatically, and to provide an accelerator system emulator for operator training.

Modeling and Inverse Modeling

There are two ways to use GLAD in the commissioning and operation of a beam line: Modeling and Inverse Modeling. In Modeling, GLAD computes the effect of the value of beam line parameters on the beam. In Inverse-Modeling, GLAD does exactly the opposite; it solves for the value of the beam line element parameters to best match the measured data. For example, during commissioning, Inverse Modeling is used to find the as-built model from the measured beam orbits (BPM data) and it is used to correct errors in the beam orbits during operation. Inverse modeling is also useful to restore beam parameters after a shut-down, hardware failure, or it is used to control the beam parameters in the presence of slow hardware drifts.

Analysis and Interpretation

For each data analysis application, the high-level software can be used to analyze beam data and to display the result graphically. Based on interpretation of the result, low-level software is used to implement the predicted adjustments. To validate the result, the beam parameters are remeasured. By comparing the measured data with the predicted result, the user decides what to do next. If the measured result does not match the prediction, the GLAD process (interactive look-analyze-interpret-adjust) continues until the as-built model is found and verified.

Manual and Automatic

Today a user must decide which beam parameter to measure, what procedure to use in the analysis, how to interpret the results, what to adjust, and when and where to iterate the GLAD process. In the future, AI methods can be used to perform these steps automatically to save valuable beam time. For example, automated error finding and beam control procedures can reduce the time it takes to recover the beam after a shut down or a hardware failure.

I am proposing the development of an AI tool kit called ASAP (Automatic System Analysis Program). Some of the AI tools to be included are Fuzzy Logic, Expert Systems, Neural Nets, and Genetic Algorithms. After the GLAD system is fully implemented, the user can commission or operate the accelerator system manually or automatically with the AI tools.

CURRENT GLAD APPLICATIONS

One typical demonstration of the GLAD process has been to find quadrupole magnet misalignment and BPM offset errors in the SLC damping rings and in SPEAR by analyzing orbit data with RESOLVE. For these cases, RESOLVE was linked to the SLC and the SPEAR control systems with adaptor codes. In this section, the use of RESOLVE to find magnet alignment and BPM offset errors in the SLC electron damping ring (NDR) is described. In particular, procedures and rules developed to analyze the beam orbits and to verify the predicted results are presented in this section.

Trial-and-Error and Rule-Based Procedures

Over the past fifteen years, special modeling and simulation codes have been developed at SLAC to make orbit corrections. All of these procedures work well. In particular, it is possible to use on-line Model-Based orbit correction procedures to reduce the residual closed orbit error in the NDR to less than a millimeter.

Unlike the orbit correction procedure, the beam injection procedure in the NDR is not model-based. An operator typically follows a trial-and-error procedure:

1. Kick the beam onto the axis of the ring by pulsing a kicker magnet,
2. Steer the beam manually along the ring axis using orbit correctors in the ring to establish a good first turn orbit,
3. Adjust correctors upstream of the kicker magnet to match the second turn orbit to the first turn orbit to obtain orbit closure.

In practice, both the closed orbit correction and beam injection procedures work. But there are two problems: first, the closed orbit resulting from optimizing the injection process is not the same as the closed orbit obtained from minimizing the RMS orbit error, and second, it is not possible to inject the beam onto the corrected beam orbit without beam loss. In either case, more than a dozen correctors are needed to steer the beam to the "good" injection orbit or to the "good" closed orbit (the NDR circumference is 35m).

To systematically investigate the beam injection/storage problem, a colleague (Jeff Corbett) and I measured several sets of first turn orbits with all of the horizontal correctors off. The orbit files were translated into RESOLVE format using adaptor codes. Using "Multi-Track" Analysis procedures [6], we soon found five BPMs with large offset errors and three misaligned quadrupole magnets in the NDR. In addition, we also identified three correctors that could be used to minimize the orbit errors caused by the misaligned magnets. Based on these model predictions, the following procedure was established to inject onto the horizontal machine axis:

1. Turn all horizontal correctors in the ring off.
2. Ignore the BPM readings with predicted offset errors.
3. Adjust the beam orbit and beam energy at the end of the transfer line to steer the beam onto the axis of the NDR, i.e. zero rms readings on the BPMs up to the first misaligned magnet. Look to see that the beam is deflected off-axis at the first misaligned quadrupole.
4. Once step 3 is true, adjust the first corrector to steer the beam orbit back onto the axis up to the second misaligned quadrupole. Look to see that the beam is deflected off-axis at the second misaligned quadrupole.
5. Once step 4 is true, adjust the second corrector to steer the beam orbit back onto the axis up to the third misaligned quadrupole. Look to see that the beam is deflected off-axis at the third misaligned quadrupole.
6. Once step 5 is true, adjust the third corrector to steer the beam back onto the axis. Look to see that the orbit in the second turn matches the orbit in the first turn.
7. Once step 6 is true, store beam and measure the closed orbit.

Prediction and Validation

The above procedure was given to the operators with a list of the three misaligned quadrupoles, the three chosen correctors, and the BPM offset errors. Following this procedure the operators were able to inject the beam on axis by obtaining the second turn orbit approximately equal to the first turn orbit within a few minutes. All of the steps went almost exactly as predicted. The operators thought our test was a success since it would take much more time and effort and many more correctors to accomplish the same result by trial-and-error.

Personal and Artificial Intelligence

The success of this experiment not only validated our predictions, it also confirmed the rules we developed for finding and verifying quadrupole alignment and BPM offset errors. As a result of our findings, two rules which will be used to automate the GLAD process are as follows:

1. A BPM is good if the predicted beam orbit agrees with the measured value for all tracks. A BPM is bad if the predicted beam orbit disagrees with the measured value for all tracks and the BPMs on both sides are good. If the difference between the measured value and the predicted value is the same for all tracks, then the difference is the BPM offset error.
2. If the beam is on axis (readings at the good BPMs are zero), the measured values at the bad BPMs are the offset errors.

In the case of the NDR, we found that the predicted offset errors at the bad BPMs agreed with the measured values. From this result, it is now possible to understand what the operators had to do without the knowledge of these offset errors. Since the operators normally used at least one corrector to steer the beam through the center of each bad BPM, at least five correctors had previously been used (incorrectly) to steer the beam. In addition, the operators used at least three correctors to compensate the three misaligned quadrupoles and two more to close the orbit. The total number of correctors therefore would add up to at least ten. With the knowledge of the alignment errors, the operator will be able to inject beam on axis by using only three correctors.

In addition, we noticed during the Model-Based beam injection/storage test that beam loss occurred at each turn in the extraction septum region. Since a localized loss

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S17AIA03

causes a beam centroid shift, it can be modelled as an abrupt "jump" in the beam orbit. Thus, in the presence of beam loss, the orbit of the injected beam can never be exactly equal to the closed orbit of a stored beam. These results suggest the following two "rule(s) of thumb" for injection/storage beam into a storage ring:

1. When the number of correctors are large, look for bad BPMs.
2. When the closed orbit can not be made equal to the injected orbit, look for beam loss during injection.

PRESENT AND FUTURE

Over the past few years, RESOLVE has been used to analyze beam trajectory data to find various types of errors in beam line elements and beam monitors at PEP, SLC, and SPEAR. In particular, procedures have been developed and tested to find dipole, quadrupole, sextupole, and rf cavity field strength errors, as well as displacement and rotational errors. Similarly, procedures have been developed to find BPM sensitivity and offset errors, and physical aperture restrictions. The development of these procedures is based on the knowledge of accelerator physics and ability to translate them into "if A then B" rules for data analysis.

Today, it is possible to compile these rules into an expert system to automate data analysis. [10] It is also possible to automate any rule-based procedure (such as the injection procedure described in the previous section) using an expert system.

In the future, other AI tools such as Fuzzy Logic, Neural Nets, and Genetic Algorithms can also be used for accelerator control. For example, Fuzzy Logic can be used for interpreting the result of the analysis since the rules may be more qualitative than quantitative (Fuzzy Rules). Neural Nets can be used to recognize errors or to handle exceptions. Adaptive feedback/correction systems can be developed using Neural Net models [11]. Finally, Genetic Algorithms can be used to train Neural Networks or to search for optimal solutions.

The development of GLAD, a Generic Lattice Debugger System, will allow us to analyze beam data with any modeling or simulation code available. In addition, GLAD will be the natural step toward developing rules and procedures for accelerator commissioning and operation, toward implementing AI methods, and toward developing automated rule-based procedures.

In conclusion, experience in accelerator control has validated a popular saying:

> *An ounce of (waste) prevention is worth more than a pound of (unnecessary) cure,*

which also tells us that,

> *An ounce of knowledge of the cause of error is worth more than a pound of correction on the effect.*

REFERENCES

[1] M. Lee, Y. Zambre, W. Corbett, "Accelerator Simulation Using Computers," SLAC-PUB-5701, August 1991.
[2] M.D. Woodley, "Control of Machine Functions or Transport Systems," in *IEEE Trans. NS-30:no. 4, Aug 1983* SLAC-PUB-3086.
[3] R. Servranckx et al., "Users Guide to the Program DIMAD," SLAC-PUB-0285, May 1986.
[4] H. Grote, F. Christoph, "The MAD Program (Methodical Accelerator Design) Version 8.4: User's Reference Manual," Iselin (CERN), CERN-SL-90-13-AP-Rev. 2, Aug 1991, 120 pp. Revised version.
[5] J. Kewisch, K.G. Steffan, "PETROS," DESY PET-76-09, DESY/Hamburg, 1976.
[6] Cf. M.J. Lee for information on RESOLVE.
[7] H. Nishimura, "TRACY: A Tool for Accelerator Design and Analysis," LBL-25236-MO, presented at European Particle Accelerator Conf., Rome, Italy, June 7-11, 1988.
[8] David C. Carey and Karl L. Brown, "New Features in Transport," SLAC-PUB-0091-Add., February 1983.
[9] S. Kleban, M.J. Lee and Y.B. Zambre, "GENI: A Graphical Environment for Model Based Control," SLAC-PUB-5126, Oct 1989. Presented at Accelerator Control: Int. Conf. on Accelerator and Large Experimental Physics Control Systems, Vancouver, Canada, Oct. 30-Nov. 3, 1989. In Nucl. Instrum. Math. A293:475-479, 1990; also in Vancouver Accel. 1989:475-479 (Journal).
[10] M.J. Lee, S. Kleban, SLAC-PUB-4539, Feb. 1988. Presented at 1st European Particle Accelerator Conf., Rome, Italy, June 7-11, 1988.
[11] D. Nguyen et al., "Accelerator and Feedback control Simulation Using Neural Networks," SLAC-PUB-5503, May 1991. Contributed to IEEE Particle Accelerator conf., San Francisco, CA, May 6-9, 1991.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S17AIA04

Development of Operator Thinking Model and Its Application to Nuclear Reactor Plant Operation System

Tetsushi Miki, Akira Endou and Yoshiaki Himeno

Power Reactor and Nuclear Fuel Development Corporation
Oarai Engineering Center 4002 Narita, Oarai-machi, Ibaraki-ken, 311-13, Japan

ABSTRACT

At first, this paper presents the developing method of an operator thinking model and the outline of the developed model. In next, it describes the nuclear reactor plant operation system which has been developed based on this model. Finally, it has been confirmed that the method described in this paper is very effective in order to construct expert systems which replace the reactor operator's role with AI (artificial intelligence) systems.

I. INTRODUCTION

A nuclear reactor plant has the following special features. How to control and operate it are the important subjects of research and development.

(1) Because it contains a lot of radioactive substances, it would harm public in case of the accidents. Therefor, its high safety is required.
(2) Because it gives society a great deal of economic loss in case of the stop of its operation, its high reliability is required.
(3) Because it is composed of many components which have different characteristics, its dynamic behavior is very complex.

In order to control and operate a nuclear reactor plant with such features adequately, the reactor operator's role is important and his burden is heavy specially in the case of the plant anomalous states. According to past serious accidents of a nuclear plant, it proved that mis-judgement or mis-operation is one of influential factors which would harm the safety and reliability of a nuclear reactor plant. Considering information processing characteristics of man and machine, the task allocation between both is decided as follows.

(1) Man is allotted to irregular tasks which require general judgement and decision making.
(2) Machine is done to regular tasks which require high speed processing.

In a current nuclear reactor plant, man takes the initiative of control and operation, and machine supports him. Therefor, various operator support systems are under development and some of them are applied to in-service real reactor plants.[1]

In order to improve further the reliability of a nuclear reactor plant, it is necessary to reduce occurrence probability of human error by replacing the reactor operator's role with the AI system. Such a plant called an autonomous one is under research and development. [2][3] In order to realize this plant, it is necessary to define a framework of the knowledge base and inference mechanisms of the AI system. One effective method would be to develop the operator thinking model and to utilize it. Based on this motivation, operator's thinking process and decision making process in the case of the plant anomalous states were studied using the full scope operator training simulator for "JOYO", the first experimental fast breeder reactor in Japan. In next, the operator thinking model was developed based on the experimental results. [4]

Still more, a nuclear reactor plant emergency operation system has been developed based on the above model. This system is an expert system which substitutes the operator's action to prevent a trip and maintain the safety of a plant in case of emergency.

II. DEVELOPMENT OF OPERATOR THINKING MODEL

At fist, the developing method of an operator thinking model is presented. In next, findings obtained by experiment and the developed thinking model are described in brief.

A. Method for Developing Operator Thinking Model

A.1 Experiment Condition

(1) Object plant : Experimental fast breeder reactor "JOYO"
(2) Simulator to be used : The "JOYO" full scope operator training simulator
(3) Experiment case

In order to attain our experiment purpose, malfunctions which satisfy the following conditions were selected.

1) They are able to be simulated by the "JOYO" training simulator.
2) They are so complex as an expert operator must think and judge.
3) They are not so complex as an expert operator cannot diagnose at all, for example, too multiple contingent malfunctions.

A.2 Simulator Experiment

The outline of typical experiment case is shown as follows.

(1) Object persons : One operator and one supervisor
(2) Selected malfunction

- Sodium leakage from the main primary B loop
- Failure that sodium leakage sensors do not operate
- Failure that sensors which detect the difference of rotation speed between A and B primary circulation pumps do not operate

A.3 Outline of Tasks for Development

Tasks for developing an operator thinking model are composed of the excusion of experiment and carried out after it.

Experimental arrangement around the training simulator is shown in Figure 1.

(1) Collection of data

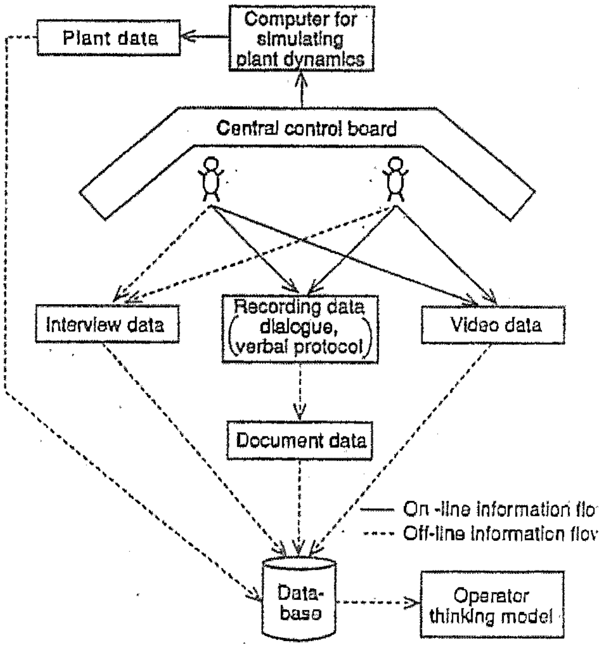


Figure 1. Experimental arrangement and data acquisition.

The following data were collected.
1) Record about an operator's verbal protocol and dialogue
2) Video camera recordings about an operator's behavior and changes in indicators on central control board of simulator
3) Analog trend data of plant dynamics
(2) Interview after experiment

Verbal protocol and dialogue data are complemented by interviewing an operator directly.

(3) Transforming record into document data

Record such as verbal protocol, interview data, etc. is transformed into document data (raw data).

(4) Making analysis data

An operator's thinking process is analyzed on the basis of raw data, and then analysis data were made.

(5) Making operator thinking model

The model which expresses universally an operator's thinking process is made by generalizing synthetically the above analysis data.

A.4 Format of Document Data

The following three document sheets are used during the development of an operator thinking model.

(1) Interview sheet
- Time when the object events of interview have occurred
- Content of interview

(2) Verbal protocol sheet
- Time when the object voice has been produced
- Content of voice

(3) Thinking process analysis sheet

Both raw data which contains the above (1),(2) and analysis data which is made on the basis of raw data are described according to the sheet shown in Table 1.

Table 1 Format of thinking process analysis sheet

Classification	Raw data					Analysis data				
Name	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)

(1) Time when events have occurred
(2) Analog trend data of plant dynamics
(3) Display information
(4) Operator's behavior
(5) Verbal protocol
(6) Segment of verbal protocol
(7) Analysis results of thinking process
(8) Recognition information of plant state
(9) Knowledge stored in long-term memory
(10) Information stored in short-menory

A.5 Analysis Method of Thinking Process Data

(1) The collected data are transformed into document data and are arranged according to the format described in Table 1.
(2) The verbal protocol data are decided into segments (the minimum units which have meaning).
(3) The macro-structure of thinking process is identified through classifying segments into basic thinking elements.
(4) The following knowledge and information used in basic thinking elements are clarified.
1) Knowledge stored in long-term memory
This knowledge is possessed by an operator before simulator experiment
2) Information stored in short-term memory
This information is memorized by an operator after the start of simulator experiment.
3) Recognition information of plant
The information flow in operator is shown in Figure 2. Think-

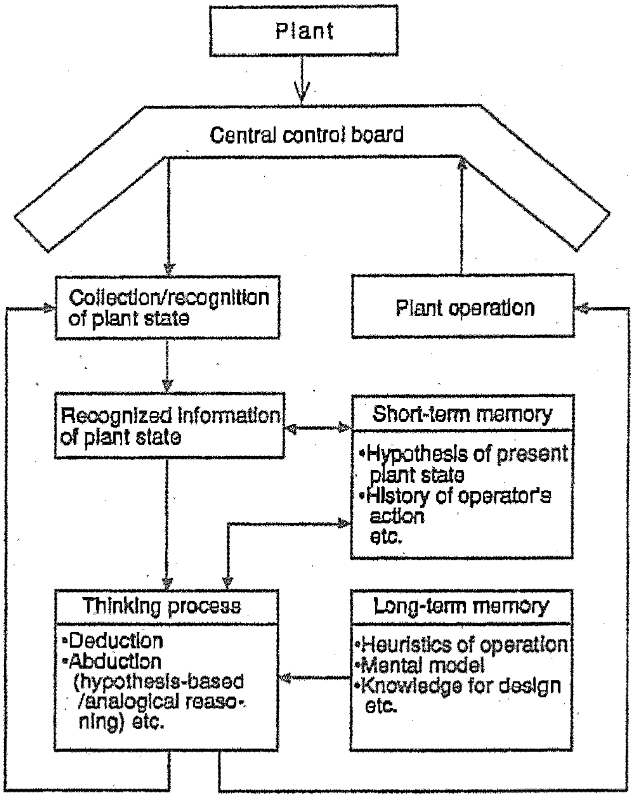


Figure 2. Information flow in operator.

ing and judging by the use of the above knowledge and information, an operator operates his plant and collects plant information.

(5) Analysis results of thinking process are arranged according to the sheet shown in Table 1.

B. Findings Obtained by Experiment

The findings were made clear in regard to the operator's procedures for decision making and action.

(1) When an operator encounters a complicated anomalous state, he acts based on his knowledge.

(2) An operator diagnoses the current plant conditions and makes his decision mainly based on hypothesis-based reasoning.

(3) When an operator cannot make suitable hypothesis only based on his shallow knowledge concerning the current plant conditions, he tries to use deep knowledge. The hypothesis is composed of the primary cause/degree/propagation of anomaly.

(4) An operator understands the relationship between goals and means to attain them in the plant on the basis of the mental model that is hierarchically composed of the operation goals and means.

(5) An operator monitors the plant conditions periodically. In the case when he faced the states of emergency, he carries out operation action against them preferentially.

C. Developed Operator Thinking Model

Based on the above findings, the thinking model was developed as shown in Figure 3. In the model, thinking and decision making procedure is as follows.

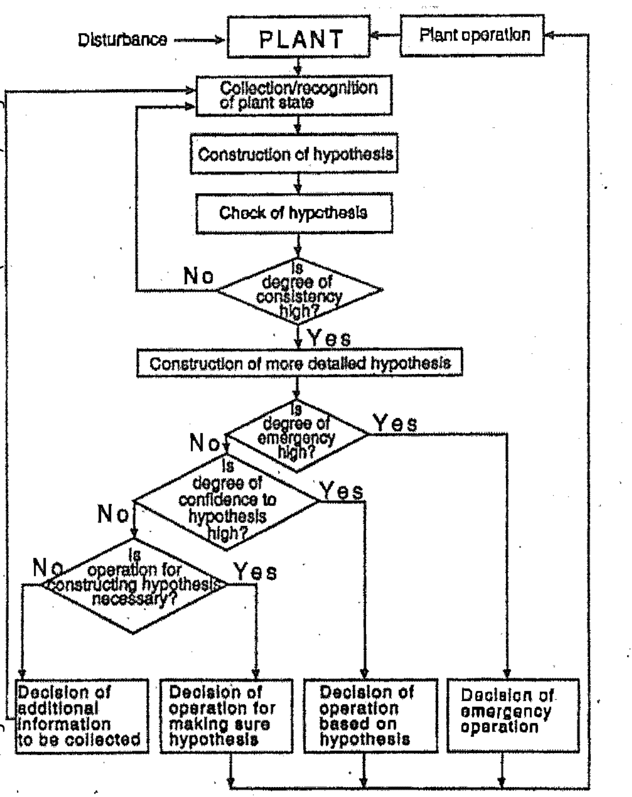


Figure 3. Developed operator thinking model.

At first, an operator actively collects and recognizes the plant information. Thereafter, he constructs a hypothesis concerning the current plant state then checks the consistency between the newly identified plant state and the hypothesis. If he recognizes that the consistency is high, he makes more detailed hypothesis. Otherwise, he makes new hypothesis. In the later portion of the procedure, he makes decision to take action depending on degrees of emergency and his confidence concerning the hypothesis. Possible actions he may take are collection of additional information, operation to make sure the hypothesis, operation based on the hypothesis, emergency operation, and so on.

It is desirable to decide plant operations based on the suitable hypothesis. But an operator decides emergency operations according to the model shown in Figure 4 in case the degree of emergency of a plant increases so rapidly that he cannot construct the suitable hypothesis. The outline of this model is as follows.

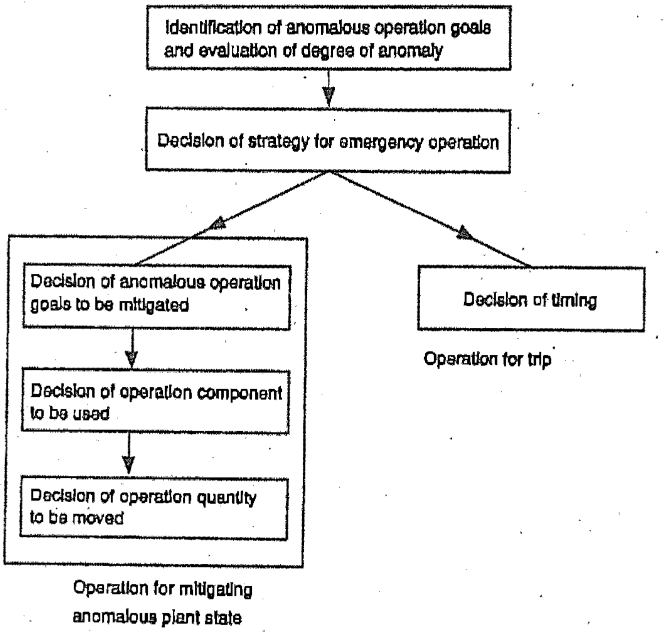


Figure 4. Decision model of emergency operation.

Based on plant display information and the mental model which is hierarchically composed of operation goals and means, an operator identifies anomalous operation goals, the degree of whose attainment are lower than threshold levels, then evaluates the degree of anomaly. In next, he selects one of the following operations and decides how to cope with anomaly newly occurred.

(1) Operation for mitigating anomalous plant state

Based on the above mental model, he decides anomalous operation goals to be mitigated, then he decides operation components to be used and operation quantity to be moved.

(2) Operation for trip

He decides the timing of trip if needed.

III. EMERGENCY OPERATION SYSTEM BASED ON OPERATOR THINKING MODEL

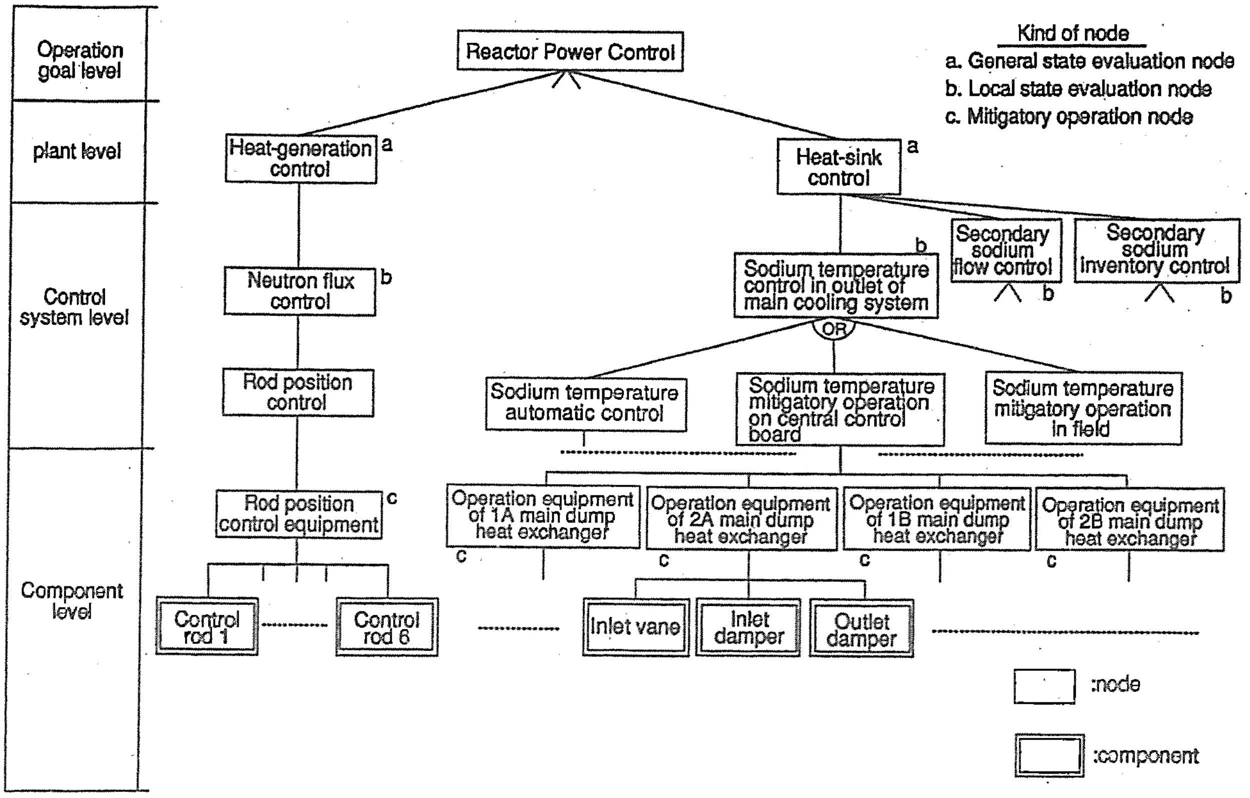


Figure 5. Partial example of operation goal network.

At first, the important knowledge base used in the emergency operation system is presented. In next, the functional constitution of this system is explained. Finally, the outline of the trial system is described.

A. Operation Goal Network

Operation goal network which corresponds to the mental model described in the preceding chapter is constructed based on operation manual and the function and structure data of a plant. It has a hierarchical structure, for nodes in upper levels show more general operation goals, on the other hand, nodes in lower levels show more concrete operation goals or means which attain more general goals. A partial example of a operation goal network whose object plant is "JOYO" and whose final goal is "reactor power control" is shown in Figure 5. This network is the important knowledge base of the emergency operation system. According to type of node, each node has some of the following information necessary to decide emergency operation.

(1) Information concerning own node, upper and lower adjacent node

(2) Information to evaluate the degree of anomaly and emergency in plant

(3) Information to calculate operation quantity

(4) Information to discriminate subsystem

(5) Information concerning operation component to be used.

B. Functional Constitution of Emergency Operation System

The functional constitution of this system is shown in Figure 6. It has a hierarchical structure which are composed of a plant monitoring system at the highest level and other subsystems at lower levels. The outline of the subsystems are as follows.

(1) Plant monitoring system

This system which is periodically activated monitors the state of a plant and decides whether emergency operations are to be carried out at once.

1) The degree of attainment of general state evaluation node is calculated according to the following steps.

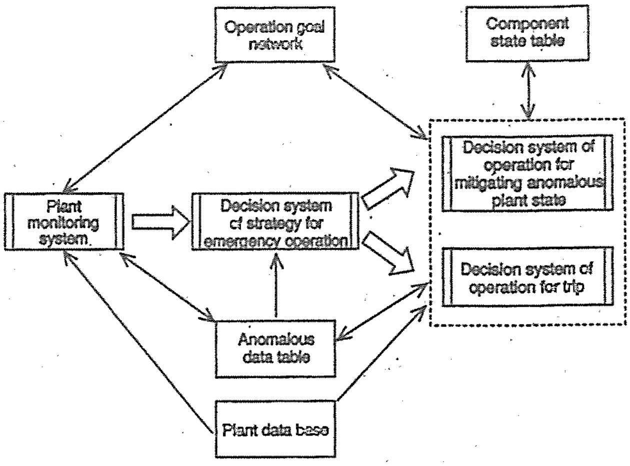


Figure 6. Functional constitution of emergency operation system.

a. The plant evaluation index which evaluates the state of a node is calculated.

This index is expressed by a function whose variables are plant data. For example, the state of node which shows "heat-sink control" is evaluated by the quantity of heat removal by a main dump heat exchanger. The quantity is calculated by multiplying a sodium flow rate by the difference between inlet and outlet enthalpy of a main dump heat exchanger.

b. The relative value x of plant evaluation index is defined as follows.

$$x=(x1-x2)/x2 \qquad (1)$$

x1: current value of plant evaluation index
x2: normal value of plant evaluation index

c. The function value f(x) is calculated using the state evaluation function shown in Figure 7.

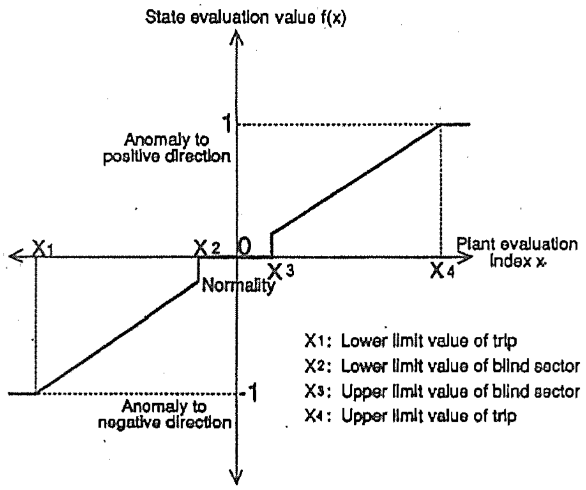


Figure 7. State evaluation function.

d. The degree of attainment DA is calculated by the following equation.

$$DA=1.0-|f(x)| \qquad (2)$$

2) Only if anomalous nodes are found, the degree of emergency is evaluated, otherwise, nothing is carried out until next calculation time.

3) Using rules which express the relationship between the degree of emergency and its influence factors, such as, the degree of attainment, its differential value, degree of importance and sensitivity to disturbance, the degree of emergency is estimated by fuzzy reasoning.

4) Based on evaluation results, it decides whether emergency operations are to be carried out at once.

(2) Decision system of strategy for emergency operation

This system decides the subsystem to be activated next from the following subsystems.

1) Decision system of operation for mitigating anomalous plant state

2) Decision system of operation for trip

Further, it decides how to cope with anomaly newly occurred.

(3) Decision system of operation for mitigating anomalous plant state

Based on plant data and operation goal network, this system decides operations for mitigating anomalous plant state in order to prevent a trip and maintain the safety of a plant.

1) Decision of operation component to be used

Using operation goal network, this system searches effective operations to mitigate anomalous plant state according to the following steps. Thereafter, it checks whether candidate operations satisfy given conditions and decides the operation to be carried out.

a. It starts searching from general state evaluation nodes indicating anomaly.

b. "AND" connection of operation goal network shows the relationship between an upper node and lower adjacent nodes all of that are required to be normal in order to attain the goal of an upper node. Because normal nodes are not necessary to be mitigated, mitigatory operation nodes are exist only under anomalous nodes. Therefore, it evaluates local state evaluation nodes and identifies anomalous subsystems which belong to these nodes. Finally, all flags which correspond to anomalous subsystems are changed from "off" to "on".

c. "OR" connection shows the relationship between an upper node and lower adjacent nodes all of that are alternative means to have the ability to attain the goal of an upper node. Mitigatory operation nodes are exist under all lower nodes which are available. All flags which correspond to available nodes are changed from "off" to "on". Thereafter, the preferential order of their nodes is decided using rules which are generated by operation manual and know-how.

d.Steps b. and c. are carried out for all nodes whose flags are "on", until mitigatory operation nodes are found.

2)Decision of operation quantity to be moved

For the combination of mitigatory operations which are most preferential, it decides operation quantity which can prevent a trip and maintain the safety of a plant by mitigating anomalous plant state. If suitable operation quantity cannot be found, different combination is tried repeatedly.

a. Identification of the combination of mitigatory operations which are most preferential

b. Decision of operation quantity

In order to mitigate anomalous plant states, the stepwise input functions Uj(t) of mitigatory operation node j(j=1~j max) must be decided so as to satisfy the following constraint equations for i=1~i max.

$$X_{imin} \leqq X_i(t) + \sum_{j=1\sim jmax} \sum_{k=1\sim imax} \delta_{i,k} \cdot U_j(t) \cdot S_{j,k}(t) \leqq X_{imax} \qquad (3)$$

t:Elapsed time from starting point
i:Number of local state evaluation node which is anomalous
j:Number of mitigatory operation node which is used for mitigating anomalous node
Sj,k(t):Operation influence function to node K by operation of minigatory operation node j(This function expresses the analog trend of the change of local state evaluation node K influenced by a stepwise operation of node j, when the plant is normal and in 100% power.)
Xi(t):Prediction function of plant variable which corresponds which anomalous node (This function is obtained by interpolating the past value of Xi and used for prediction of the future value.)
Ximin:Lower limit value which Xi must not violate in order to maintain the trip margin or plant safety
Ximax:Upper limit value which Xi must not violate
δ i,k:Sign which shows 1 if and only if i equals k, otherwise, shows o

c. Decision of detailed operation way

If operation quantity which satisfies Eq. (3) cannot be found,the above calculation is continued for different combination. Otherwise, detailed operation way is decided based on operation manual, available operation components and their preferential order, etc..

3) Confirmation of results of operation

It confirms whether decided operations are carried out correctly.

(4)Decision system of operation for trip

Based on plant monitoring data, etc., it decides the timing of trip in case the degree of emergency of a plant increases so rapidly that it cannot decide suitable emergency operations.

C.Trial System

The trial system which has the above functions has been constructed. The outline of this system is as follows.

(1)Computer
1)Types of computer:SUN-4 workstation
2)Language:C-language
(2)knowledge base

The outline of the operation goal network which is the important knowledge base used in this system is as follows.

1)Number of node:25
2)Number of state evaluation function:10
3)Number of fuzzy function to evaluate the degree of emergency:50
4)Number of operation influence function:50
(3)Data base
1)Number of data composing plant data base:50
2)Number of data stored in anomalous data table:10
3)Number of data stored in component data table:50

Based on results of operation of this system, it has been clarified that it can decide emergency operations in real time.

IV. CONCLUSION

The operator thinking model was developed using the operator training simulator, then and the emergency operation system was constructed based on this model. Based on results of operation of this system, it has been confirmed that the method described in this paper is very effective in order to construct an expert system which can replace the reactor operator's role with AI system. Application of this model to developing an autonomous plant is intended by refining the model and programming other part of it.

V. REFERENCES

[1] Y.Himeno, T.Nakamura, S.Terunuma and T.Furubayashi, "Improvement of man-machine interaction by artificial intelligence for advanced reactors," Proc. of SMiRT 11th. Post-Conference Seminar on Probabilistic Safety Assessment Methodology, Kyoto, Japan, 1991.

[2] T.Miki and K.Tamayama, "Conceptual research of autonomous nuclear power plants," Power Plant Dynamics, Control & Testing Symposium, Knoxville, Tennessee, USA, 1989.

[3] K.Watanabe, T.Miki, A.Endou and Y.Himeno, "Development plan of operation system for autonomous nuclear plant," International Conference on Fast Reactors and Related Fuel Cycles, FR'91, Kyoto, Japan, 1991.

[4] T.Miki and K.Tamayama, "Development of a nuclear plant operator's thought model by the use of a training simulator," Proc. of Topical Meeting on Advances in Human Factors Research on Man/Computer Interactions : Nuclear and Beyond, Nashville, Tennessee, USA, 1990.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S18BPA01

Operational Decoupling in the SSC Collider

George Bourianoff
Accelerator Design and Operations Division
Superconducting Super Collider Laboratory*
2550 Beckleymeade Avenue
Dallas, TX 75237

I. INTRODUCTION

This paper will summarize a recent study of the effects and correction of linear coupling in the Superconducting Super Collider (SSC) lattice. There are several aspects of the SSC lattice that make direct extrapolation of techniques used on existing machines unreliable. The most obvious aspect of the SSC which departs from previous experience is the small dynamic aperture which lies well within the beampipe. A second aspect is the existence of long arcs with low superperiodicity which allow various sources of skew quadrupole to accumulate to large, and, perhaps, nonlinear values. A third aspect is the relatively large value of systematic skew quadrupole error in the main dipoles. This results from asymmetric placement of the cold mass in the cryostat.

Coupling must be considered harmful if it leads to irreversible emittance blow-up, a decrease in the dynamic aperture, or inoperability of the machine. These negative effects are generally related to coupling terms that accumulate to large and, hence, nonlinear values prior to correction. The harmful effects can also be caused by the linearly coupled orbits interacting with high-order multipole fields that exist in the other magnets.

The errors that lead to linear coupling are well known. They are systematic and random skew quadrupole error fields in the other magnetic elements, angular alignment errors in the quadrupoles and feeddown from the sextupole fields associated with chromaticity correction, and persistent current fields in the dipoles. A study of the relative importance of the various coupling terms for a simplified SSC lattice is contained in Reference 1 (SSC-N-93 by Richard Talman).

The traditional way of correcting linear coupling is to use two families of skew quads and adjust them so that the separation between the two betatron tunes is minimized. This is a global scheme which is sensitive to only the betatron tunes which clearly are global quantities. Another traditional method is to focus on the sum and difference resonances and use two families to skew quads to correct them. Sometimes both methods are used, which requires four families of skew quads.

The method being proposed for the SSC is intrinsically different and amounts to decoupling one local section at a time. The mathematics required to do this are described in the next section. The primary motivation for local decoupling is that the errors are not allowed to accumulate and reach the level at which irreversible nonlinear mixing occurs. Local decoupling requires a measure of the local effect of the errors rather than the global effect of the errors. The quantity that goes directly into the decoupling calculations is simply the ratio of the out-of-plane tune amplitude to the in-plane tune amplitude. This quantity is directly measurable and has in fact been measured on the HERA proton ring and the Fermilab main ring.

Section 3 contains simulation results of the SSC collider with all known errors included and a full simulation of the correction process. It was found that 46 pairs of independently controlled skew quads are adequate to obtain a reasonably decoupled machine with a reasonable dynamic aperture. It was likewise determined that 16 pairs of skew quads are not sufficient to decouple the machine at injection optics. The minimum acceptable number and optimum placement of skew quads is a matter of continuing study.

A. Decoupling Formulation

The analytic formulation of the local decoupling algorithm is contained in Reference 2, Talman's discussion of single particle motion. A few of the key results will be repeated here for the sake of completeness.

The propagation of the four-dimensional phase-space vector X from point s_0 to s_1 is written in terms of the four-dimensional transfer matrix M shown below:

$$X_1 = M_{10}\, X_0 \quad . \tag{1}$$

The localized transfer matrix M_{10} may be written in terms of the once-around transfer matrixes M_1 and M_0 at locations s_1 and s_0, respectively. The relationship is given as

$$M_1 = M_{10}\, M_0\, \overline{M_{10}}\ , \tag{2}$$

where the bar indicates the symplectic inverse of the localized transfer matrix. The approach is to block diagonalize the once-around transfer matrixes at a sequence of points $s_0, s_1, \ldots, s_n$. The procedure necessary to block diagonalize each individual transfer matrix starts by breaking up the 4 × 4 matrix into four 2 × 2 matrices denoted by A, B, C, and D. One then forms linear combinations of the B and C submatrices responsible for coupling:

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} . \tag{3}$$

---

*Operated by the Universities Research Association, Inc., for the U.S. Department of Energy under Contract No. DE-AC35-89ER40486.

These are called R_A and R_D and are given by these expressions:

$$R_A = \frac{C + \overline{B}}{\Lambda_A - tr\,D}\,, \qquad R_D = \frac{B + \overline{C}}{\Lambda_D - tr\,A}\,. \tag{4}$$

It can be shown that the matrixes R_A and R_D are not independent of each other and that only four of the eight terms in the two matrixes are in fact independent. It is possible to form two parameters, e_A and e_D, which are physically observable and can be minimized to minimize the coupling. The expression for e_A is given below with a completely analogous expression for e_D:

$$e_A^2 = \left[R_{A11} + \left(\frac{\alpha_A}{\beta_A} R_{A12}\right)\right]^2 + \left(\frac{R_{A12}}{\beta_A}\right)^2 . \tag{5}$$

The value of eA **2 is equal to the ratio of the out-of-plane tune amplitude to the in-plane tune amplitude and is a physically measurable quantity. Section 4 describes measurements of this quantity made in the HERA proton ring. The decoupling process consists of forming a weighted sum of the e_A terms at the position of every instrumented BPM, calculating the set of skew quad corrector strengths necessary to minimize the sum, and applying these strengths at the skew quad locations. It should be noted that this method has sufficient information to set each skew quad independently, and it is not necessary to connect the skews in two or four families.

B. Decoupling Studies at the SSC

The specific problem of decoupling the SSC lattice has been studied using the accelerator simulation code described in Reference 3 (Schachinger and Talman). The specific issue has been to investigate the possible planments, patterns, quantity, and strengths of skew quads necessary to decouple the collider.

The quantities used to parameterize the decoupling process are the decoupling badness function defined above and the peak and average values of the vertical dispersion. A brief summary of results is contained in Table 1.

The study concluded that the collider can be satisfactorily decoupled in the presence of all known coupling terms if the tunes are separated by an integer difference of at least 1. It is necessary to use approximately 100 independently powered skew quads located in cell centers which have a strength of (GL) of 35 Tesla. Alternatively, 56 skew quads at the location of the missing dipoles will decouple the lattice, but a peak strength of 67 Tesla appears to be required.

The optimum location, number, and placement of skew quad correctors are matters of continuing study. The ultimate choice of corrector scheme will involve a consideration of the dynamic aperture in addition to the local badness and vertical dispersion criteria. However, these last two parameters provide a quick means of making preliminary comparisons of different corrector layouts.

C. Experimental Measurements at HERA Proton Ring

The local decoupling technique described in the previous sections requires as input the local decoupling badness function, which is defined to be the ratio of betatron tune amplitudes measured at many places around the accelerator. The ability to measure the decoupling badness in a real operational environment is critical for the local decoupling method to work. An experimental measurement of this quantity was therefore carried out on the HERA proton ring. The experiment consisted of capturing turn-by-turn BPM data from each of the 137 monitors in each plane (274 monitors total) at injection. The data were analyzed using Fourier transform techniques, and the horizontal and vertical betatron tunes were determined. The amplitude of the vertical betatron tune line in the horizontal BPM data was extracted, as was the amplitude of the horizontal tune line (which was, of course, dominant). The transformed BPM data for the horizontal and vertical planes are shown in Figures 1 and 2, respectively. The ratio of the small-to-large tune amplitudes at each BPM is given in Figures 3 and 4.

Table 1

Number	Badness	Vertical Dispersion	Comments
			small al (0.04 units) GREV5 lattice
32	4.69	0.4 m rms	$\mu_x = 123.285$ $\mu_y = 124.265$
32	0.059	1.1 m rms	$\mu_x = 123.285$ $\mu_y = 125.265$
			injection energy 1.8 TEV
32	NA	NA	no solution unstable optics
			al = 0.3, 10F lattice 20 TEV
92	0.0029	0.8 m rms 2.2 m peak	one pair of skew quads every 8 cells
100	0.0031	0.4 m rms 1.6 m peak	minimum eta placement
200	0.001	0.2 m rms 0.8 m peak	one pair of skew quads every 4 cells
47	0.03	0.3 m rms 0.8 m peak	one quad in missing dipole 500 detectors

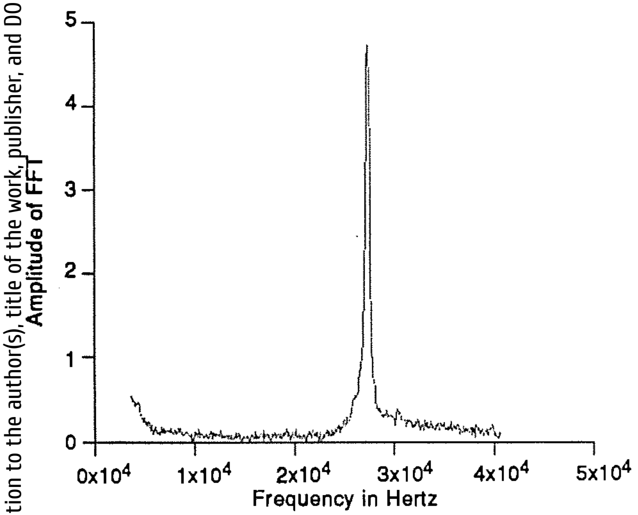


Figure 1. Horizontal Data.

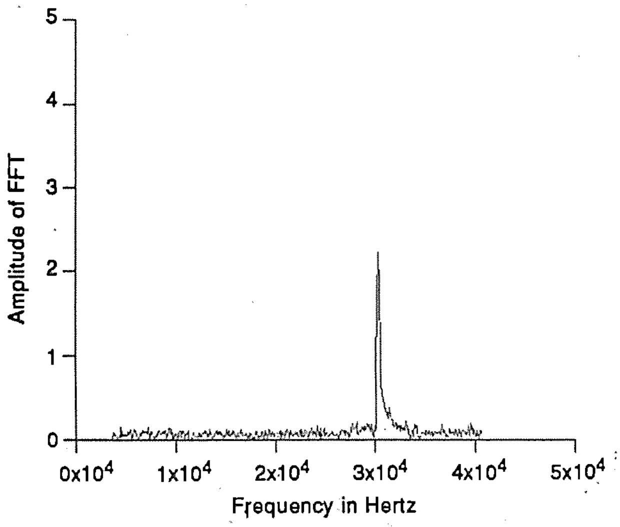


Figure 2. Vertical Data.

The horizontal coupling badness has a maximum value at the location of the skew quad pair in the IR sections. However, the RMS value is less than the corresponding function in the vertical plane, indicating that the global decoupling will put a large local perturbation into the optical functions while reducing the overall coupling. The magnitude of the coupling functions are proportional to the tilt of the local eigenplanes.

The measurement of the local coupling functions on an operating accelerator shows that the quantities necessary for local decoupling can be experimentally measured. It remains to be demonstrated that these values can be used to locally set skew quadrupole strengths.

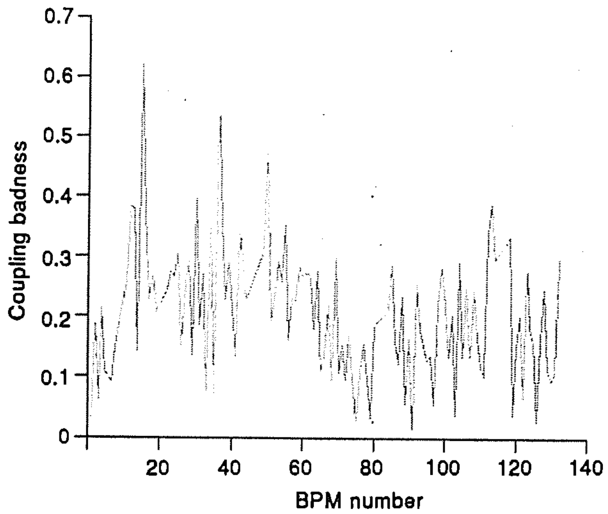


Figure 3. Horizontal Coupling Badness.

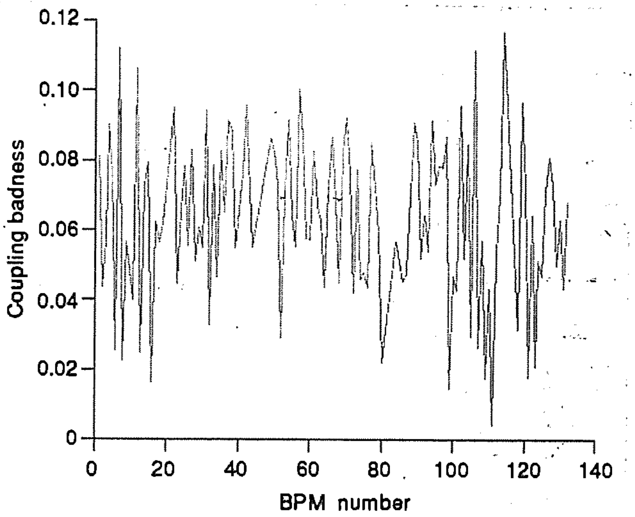


Figure 4. Vertical Coupling Badness.

II. CONCLUSIONS

This paper has shown results indicating that it will be possible to locally decouple the SSC lattice using a local decoupling algorithm. The mathematical basis of the algorithm was described briefly, and references were given for a complete description. Experimental measurement of the local coupling function required by the algorithm has been demonstrated on the HERA proton ring.

III. REFERENCES

[1] Richard Talman, Coupling Effects in the SSC Regular Arcs, SSC-N-93, December 1985.
[2] Richard Talman, "Transverse Motion of Single Particles in Accelerators," *AIP Conference Proceedings*, No. 184, 1987, p. 190.
[3] L. Schachinger and R. Talman, "TEAPOT: A Thin-Element Accelerator Program for Optics and Tracking," *Particle Accelerators*, 1987, Vol. **22**, pp 35-56.

Frequency Domain Analyses of Schottky Signals Using a VME Based Data Server and a Workstation Client

A. Chapman-Hatchett, V. Chohan, I. Deloose, F. Pedersen
CERN, PS Division, CH-1211 Geneva 23

Abstract

Schottky signals are extensively used for observation, setting-up and operation of CERN's Antiproton rings, namely the AC, the AA and LEAR. Measurement of these signals is, at present, carried out by a series of commercial instruments. These instruments have to be individually controlled and read by each application program. The operational use of the system is limited by the capabilities of the individual instruments. The first objective for the new system was to provide, as far as possible, a true "server". The "client" application program simply requests the data it requires. It is then supplied with measured and processed data. This provides the operator with a fast response by having ready processed data always available. Our second goal was to make the system operationally simple, with multiple windows and presentation on a single screen. This paper discusses some aspects of this implementation and applications for the antiproton production, collection, and storage rings.

INTRODUCTION

The system is to be used on the CERN Antiproton Collector(AC) and Antiproton Accumulator (AA) rings. The Antiprotons are first created by directing a 26 GeV proton beam onto a production target. The antiprotons are then captured, injected and stochastically cooled in the AC ring.

After cooling they are transferred to the AA ring where they are stored, and continuously cooled until needed by the physics programme. This is a multistep process involving rf bunch rotation and debunching, stochastic cooling in all three planes, rf recapture in the AC and bucket-to-bucket transfer to the AA, pre-cooling in the AA, rf capture in the AA and transfer to the stack-tail region and stack-tail cooling as well as core cooling for long-term storage in the AA. To have any yield at the end of this process, all the systems have to be set up to run with very high efficiency. In addition, due to the limited available cooling power, beam blow-up must be avoided at every step. Similarly, non-destructive beam diagnostic measurements have to be made, i.e. without disturbing and blowing up the beams. The technique chosen was spectrum analysis of the Schottky signals produced by the beams [1, 2]. By using appropriately positioned pickups and producing spectra of the revolution frequency sidebands, the beam profile in the horizontal, vertical, and longitudinal planes can be deduced. If the sensitivity of the system is known, the integral of the spectrum gives the beam intensity. At present, four types of spectrum analyzers, from two different manufacturers are used. Differences between these instruments leads to very complex measurement programming. To complement or replace these instruments, a very high speed, fast Fourier transform engine is needed. The VASP16 VME module from Computer General was chosen. This has a Texas Instruments 320C25 digital signal processor, and four Zoran vector processors on the board. The board is capable of performing a 1024 point FFT in 700 µs. The VME chassis is controlled by a Motorola MVME147 board running the OS9 operating system. The Motorola 68030 CPU transfers data from the ADC's (analogue to digital converters) to the VASP16 and services requests for data from the users. The ADC's are triggered to take data by the accelerator timing system.

OBJECTIVES

Replacing these instruments with a VME system, there are four main objectives.

a) To make the results available on request. At present each instrument has to be set up and then the measurement can be made. With the VME chassis data taking and processing can take place continuously, and the results presented on demand.

b) The spectrum analyzers have some real time constraints, they normally expect to acquire data, then process it, then display it. The VME system can acquire data at times fixed by the machine cycle, and fit in the processing between data acquisition.

c) The third objective is to separate the process of data acquisition from the task of presentation. This would allow any operator to access data without interfering with another operator's measurements.

d) This system has to be integrated into the control system. By using the VME system we can produce a standard interface to the control system, with all the specific low-level software hidden from the rest of the system.

SYSTEM OVERVIEW

Figure 1 shows a generalized overview of the system.

The resonant pickups consist of two parallel plates inside the vacuum chamber, one pair for the horizontal measurement, and a pair for the vertical measurement. The signals are brought out with vacuum feedthroughs and fed into a coaxial line resonator. The sum and difference signals are made using a hybrid and the resulting signals are amplified by low-noise amplifiers. The specific characteristics and needs for the respective rings are shown below.

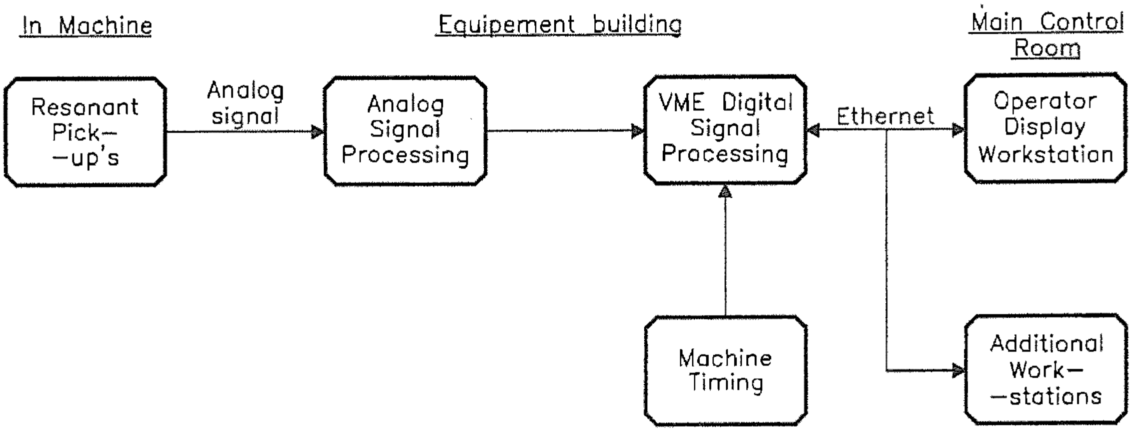


Fig. 1. System Overview

	Center freq. MHz	Bandwidth MHz	Gain dB	Rev. freq. harmonic
AC	49.2202	1.5	60	31
AA	71.9379	4.3	60	39

The system will cater for six analogue signals, i.e. the sum, horizontal, and vertical signals from both the AC and AA rings. There will be six parallel channels up to the VME chassis. The additional workstations could be in the control room, in a laboratory or even in an office, the only requirement would be the availability of access to the control system Ethernet.

ANALOGUE PROCESSING

The signals for each measurement all have similar analogue processing. The signals are first down converted using upper sideband mixers. They are then amplified and filtered by elliptic antialiasing filters, and further amplified to match the full scale range of the ADC's. Figure 2 illustrates this schematic layout.

DIGITAL PROCESSING

The acquisition is started by a timing pulse from the accelerator (AC or AA) triggering the analogue to digital converter. This acquires 8192 samples at four microsecond intervals. When the data is acquired the data reading software is flagged to copy the data to a circular buffer, reset the converter, and flag the data treatment software. The data treatment software has two options. If the data was acquired at injection, it copies 512 samples of raw data to a named data module, and performs a single FFT on the first 1024 data samples. This gives a measurement of the injection process. Data modules here imply the software structures as permitted under the OS9 operating system.

If the measurement was not at injection, it performs 16 sliding FFT's on the 8192 samples and then averages the resulting spectra. All these results are then saved in other data modules ready for any requests via the control system.

In addition the data treatment software adds to the data module any other information that might be needed by a display program, for example, gains, timing information, labels for graph axes.

COMMUNICATION AND OPERATOR INTERFACE

The communication between the VME crate (server) and the DEC-3100 workstations (clients) is established using the standard TCP/IP socket library and Ethernet interface. A user defined port number is used to access the self-written driver installed on the VME crate to serve the clients. To provide the separation between data acquisition on the server and the final display on the

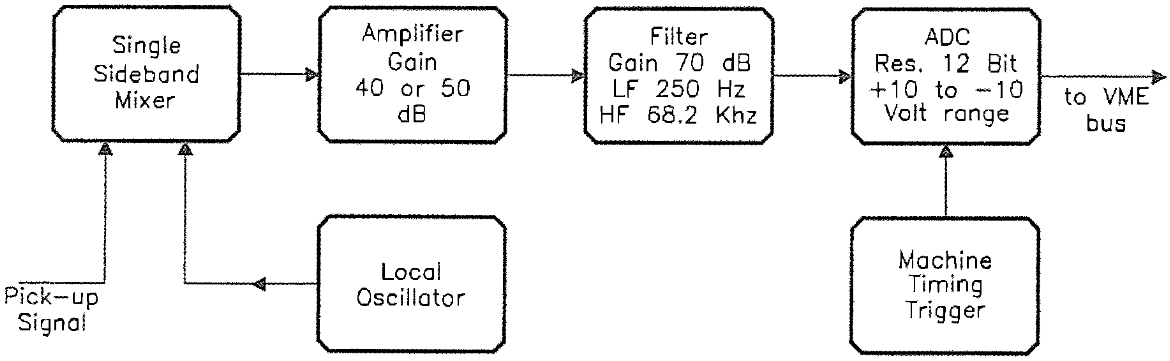


Fig. 2. Layout for analogue signal processing

client, a prototype database approach has been implemented on the server. A database file contains the names of all the data modules available to the client. This file also permits a multi-name mapping between a user preferred name and the real name of a data module.

Once the server accepts the connection request for a client through the user port, it sends all the relevant information including labels, axes information, etc. relevant to a particular data module to the client. The user selection of a named module is done through the menu-driven selection box on the workstation. This name is translated onto the real data module name and the request added to the active (hot) link table on the server. The data module header contains the date and time information. This is useful and necessary because the server task is continually scanning the date-time stamp in the headers of all the data modules in the hot link list and transferring the latest data blocks to the corresponding clients. The server task also checks if a client request has terminated and the validity of the client-server link. If the link is severed, it removes all the data modules of the interrupted client from the hot link list.

The data from each data module is displayed graphically in a window as shown in Fig. 3. For multiple data modules, a multiple set of windows are put together in a main window to permit a sequential observation of a chain of processes as applicable to the passage of a single beam shot of the antiproton collector and accumulator complex. Depending on the data, each window is individually auto-scaled and re-sized. Using the standard tools of the MOTIF tool kit, zooming, integration of values between set markers or the evaluation of co-ordinates at any one point on the graph is easily permitted.

SOME RESULTS

Figures 4 and 5 show the sum signal acquired in the AC ring at the time of injection. Figure 4 clearly shows the appearance of Schottky noise as the beam arrives into the ring while Fig. 5 illustrates the power spectrum of the same data. The spectrum is equivalent to the beam longitudinal profile. Figures 6(a) and 6(b) illustrate the beam spectrum obtained in the AC ring at two different time points. Figure 6 (a) shows the beam profile straight after the rf bunch rotation while Fig. 6(b) illustrates the longitudinal beam size, 4.5 s after injection in the AC ring, i.e. at the end of all cooling processes in the AC.

CONCLUSIONS AND FUTURE WORK

Although the system is not completely operational, it already shows sufficient promise as a good approach for the future. The aim of separating the data acquisition and display software has been achieved; after agreeing to the context of the data structures that have to be exchanged, the development of the two parts of the software took

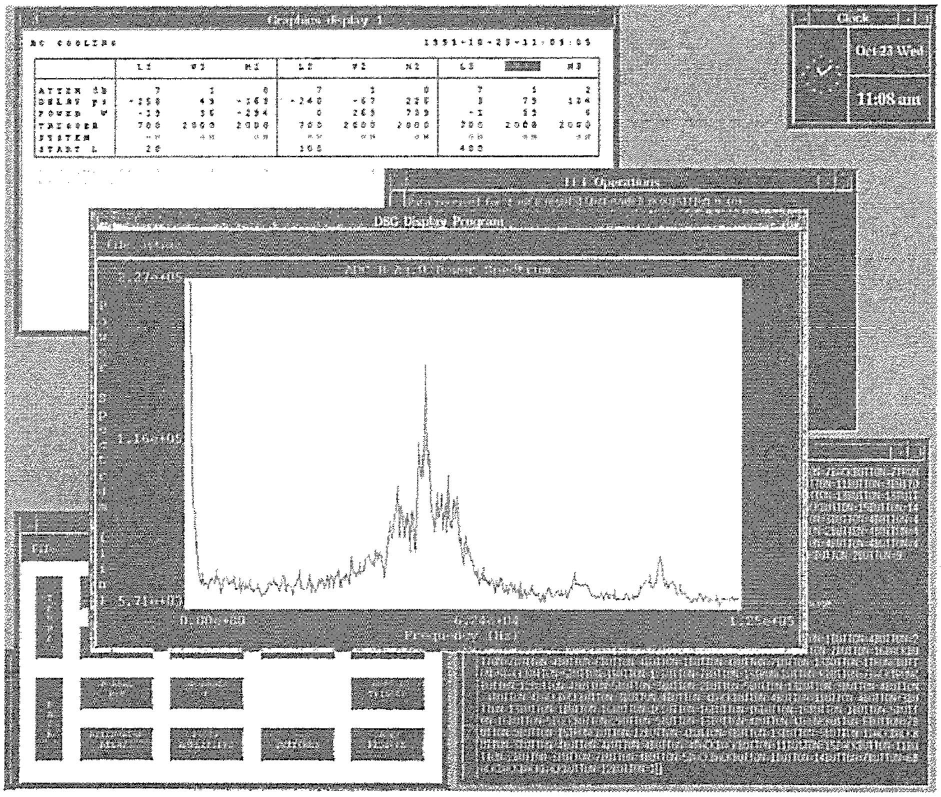


Fig. 3. Typical data module graphical window on a workstation

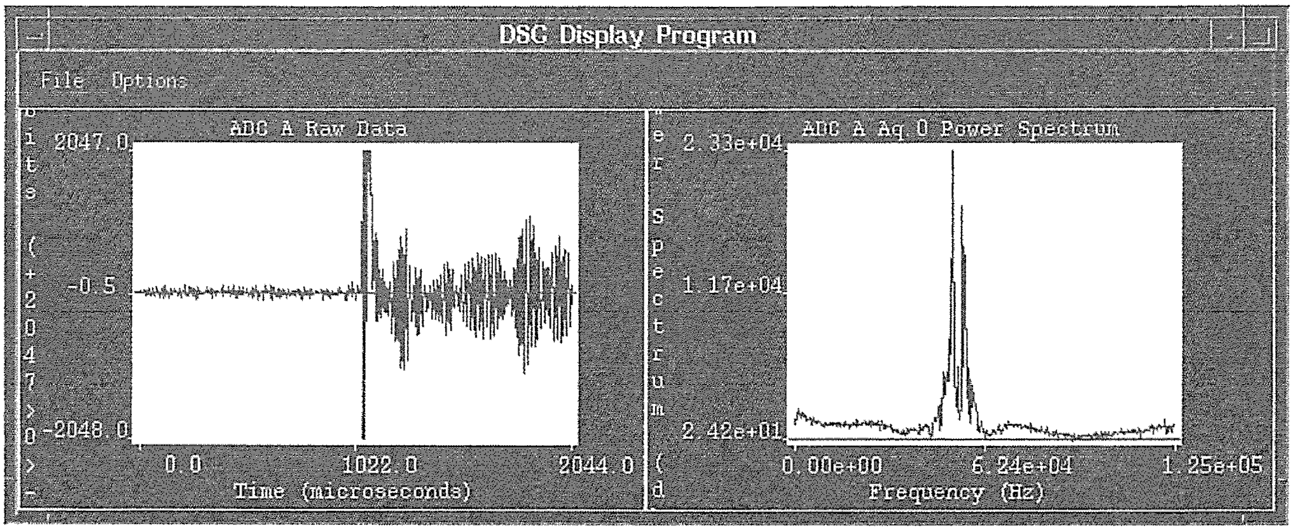


Fig. 4. Beam arrival in the AC Ring

Fig. 5. Power spectrum of the AC injected beam

place completely separately. The aim of having the data available on request from the client has been achieved. The operator can work at his own place, changing the displays as required. The system is pleasant to use, with measurements taking few seconds instead of several seconds in the old system.

At present the data processing is using a prototype program which takes approximately one second to acquire the data, make 16 fast Fourier transforms, and produce the mean of the spectra. The definitive program, when commissioned, will take about 100 ms. Current results show that the resolution needs to be doubled to give a good measurement with at least 10 data points on the fully cooled beam. Similarly, for the injected or uncooled beam an improvement in signal to noise ratio by a factor of two, would give a spectrum which would be much easier to interpret.

REFERENCES

[1] Borer J., Bramham P., Hereward et al., "Non-destructive diagnostics of coasting beams with Schottky noise", *Proc. of the IXth Int. Conf. on High Energy Accelerators*, Palo Alto, USA, May 1974, and CERN/ISR-DI/RF/74-23, 1974.

[2] van der Meer S., "Diagnostics with Schottky noise", *Proc. of Joint US-CERN School on Beam Observation, Diagnosis and Correction*, Capri, Italy, Oct. 1988, and CERN PS/88-60 (AR), 1960.

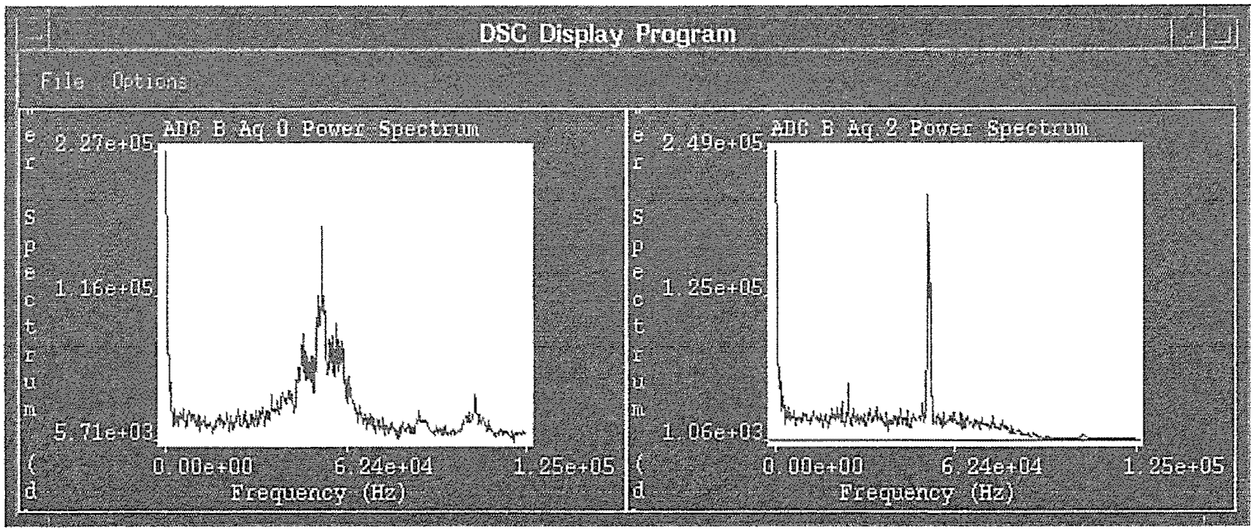


Fig. 6. (a) Beam in AC Ring before cooling and (b) after cooling for 4.5 seconds

New Controls for the CERN-PS Hadron Injection Process using Operating Tools and High-Level Accelerator Modelling Programmes

M. Arruat, M. Boutheon, L. Cons, Y. Deloose, F. Di Maio, D. Gueugnon,
R. Hoh, M. Martini, K. Priestnall, J.P. Riunaud
PS Division
CERN 1211 Geneva 23 Switzerland

Abstract

A new control system using man-machine interface tools with workstations as consoles has been successfully put into operation for the injection of hadrons in the CERN Proton Synchrotron (PS). This paper mainly focuses on specialized modelling programmes involving complex treatments for an optimum operation of the injection process. These programmes include the control of the injection timings, the measurement of the beam emittance with an estimation of how well the incoming beam is matched, and the correction of oscillations at injection. The infrastructure and the programming environment underlaying the new control system are described elsewhere[3].

The outstanding feature of the internal structure of all these modelling programmes is that they carry out three kinds of data interaction: the input, that is the measurements (e.g. beam time positions, profiles and trajectories), the physical parameters (e.g. required times for synchronization, beam emittance, beam space position and angle at injection), and the output, mainly the hardware values (e.g. preset counter settings, currents to apply to injection steering magnets).

I STRUCTURE OF HIGH-LEVEL MODELLING PROGRAMMES

A control system provides the users with centralised access to hardware control values. These values can be obtained in various ways. They can be controlled individually, for example tuning a power supply current via a knob, or globally, setting all currents of a complete beam line from a selected file. In those two cases, the final hardware values are left to the user's appreciation. Their choice is made either following the effect on the beam, as in the first case, or at the time of the file selection, as in the second. Evaluation of these control values does not require implementation of the process involved in the control system.

However, in many cases the process involved is known and several control values can be worked out from the required machine and/or beam parameters. A dedicated programme can then be used to evaluate these control values from the user's requirements. Considering the previous example of a beam line, if the relationship between the power supply currents and the energy is known, setting a complete beam line for various energies does not require as many files as energies, but only one file and a dedicated programme working out the currents from the required energy.

This trivial example can be extended to more general cases. In the example of beam steering, the trajectory of the beam in a transfer line can be taken as input, physical parameters such as angle and position of the beam at a relevant position in the line can then be worked out from this trajectory, and finally current variations in deflecting elements can be computed to steer correctly the beam in the line. More generally, controls values (output) can be computed from physical parameters (parameters relevant to the process involved) which, in turn, can be computed from measurements on the beam (input). Modelling Programmes perform such operations and take care of computations between inputs, physical parameters and outputs.

In most cases, provided they are not too large, variations of beam characteristics (ΔX, inputs) lead to variations of physical parameters (ΔP) which can be expressed in a matrix formalism. The same applies to the variations of hardware control values (ΔY, outputs) which can be expressed from variations of the physical parameters:

$$[\Delta P] = [M_1] * [\Delta X] \quad (1)$$

$$[\Delta Y] = [M_2] * [\Delta P] \quad (2)$$

Modelling Programmes can compute hardware values from physical parameters or from beam measurements, using matrix M_1 or both M_1 and M_2. If matrix M_2 is not singular, one can also work out physical parameters by reading control values. This can of course be of great help for machine tuning. In the following section we present how these considerations have been applied to the timing process of hadron injection into the PS[1].

II INJECTION TIMING CONTROLS

In the injection process, timing pulses have to be delivered to various equipments in order to trigger them correctly with respect to the incoming beam. Preset counters are interconnected in such a way that they can provide the necessary pulses, with the proper time resolution, at appropriate instants with respect to external time references. One can then define the required times as physical parameters and the output as the control values to be loaded in the various counters. Looking at the timing lay-out, the required times can be defined from the time of the External Starts and the sum of the various count times elapsed in the relevant counters. This can be expressed as the following matrix expression:

$$[\text{Required Times}] = [\text{External Starts}] + [\text{M lay-out}] * [\text{Count Times}] \qquad (3)$$

where M lay-out is a matrix reflecting how pulses proceed through the various counters. Its elements are null if the corresponding counter is not involved, 1 if the counter is involved and its time has to be added, and -1 in the opposite case. Count Times are simply obtained from the counters control values and their clock periods Tck:

$$[\text{Count Times}] = [\text{Control Values}] * [\text{Tck}] \qquad (4)$$

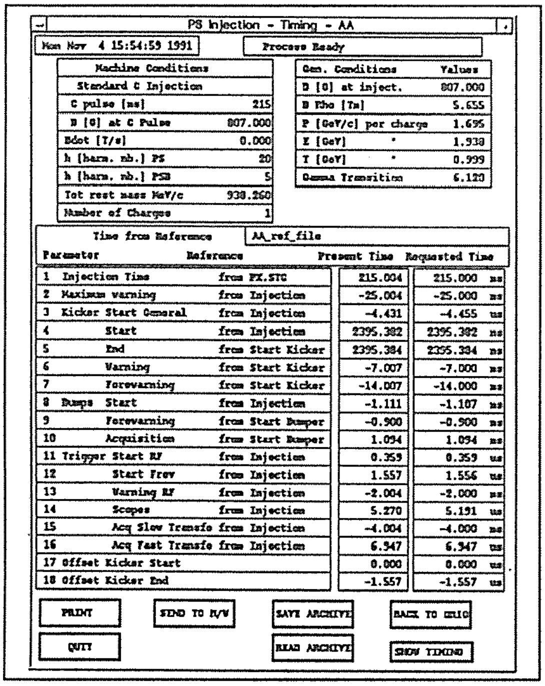
If one defines as many Required Times as counters in the lay-out, M lay-out is not singular and the control value can be deduced from the preceding expressions:

$$[\text{Count Times}] = [\text{M lay-out}]^{-1} * \{[\text{Required Times}] - \text{External starts}]\} \qquad (5)$$

$$[\text{Control Values}] = [\text{Count Times}] / [\text{Tck}] \qquad (6)$$

Clock periods Tck can be evaluated from the hardware settings (for internal clocks) or from machine parameters (for external RF or Field-derived clocks). Therefore all control values can always be worked out from the required times, in all machine conditions.

This has been applied to the PS hadron injection timing. The system layout has been translated into a matrix M lay-out to define the required times. The resulting interactive display is shown in Fig. 1.



Mon Nov 4 15:54:59 1991

Process Ready

Machine Conditions	
Standard C Injection	
C pulse [ms]	215
B [G] at C Pulse	807.000
Bdot [T/s]	0.000
h [harm. nb.] PS	20
h [harm. nb.] PSB	5
Tot rest mass MeV/c	938.260
Number of Charges	1

Gen. Conditions	Values
B [G] at inject.	807.000
B Rho [Tm]	5.655
P [GeV/c] per charge	1.695
E [GeV] *	1.938
T [GeV] *	0.999
Gamma Transition	6.120

Time from Reference AA_ref_file

Parameter	Reference	Present Time	Requested Time	
1 Injection Time	from PX.STC	215.004	215.000	ms
2 Maximum warning	from Injection	-25.004	-25.000	ms
3 Kicker Start General	from Injection	-4.431	-4.455	us
4 Start	from Injection	2395.382	2395.382	ms
5 End	from Start Kicker	2395.384	2395.384	ms
6 Warning	from Start Kicker	-7.007	-7.000	ms
7 Forewarning	from Start Kicker	-14.007	-14.000	ms
8 Dumps Start	from Injection	-1.111	-1.107	ms
9 Forewarning	from Start Dumper	-0.900	-0.900	ms
10 Acquisition	from Start Dumper	1.094	1.094	ms
11 Trigger Start RF	from Injection	0.359	0.359	us
12 Start Frev	from Injection	1.557	1.556	us
13 Warning RF	from Injection	-2.004	-2.000	ms
14 Scopes	from Injection	5.270	5.191	us
15 Acq Slow Transfo	from Injection	-4.004	-4.000	ms
16 Acq Fast Transfo	from Injection	6.347	6.347	us
17 Offset Kicker Start		0.000	0.000	us
18 Offset Kicker End		-1.557	-1.557	us

Fig. 1: Interactive display of the timing modelling programme. Present times are computed from the hardware acquisition through Eqs. (3-4). Requested times are filled in by the user and provide control values through Eqs. (5-6).

III EMITTANCE AND MATCHING MEASUREMENTS

A method for the measurement of transverse emittance in the PS and matching of the beam between the Proton Synchrotron Booster (PSB) and the PS rings was implemented in the present control system[2]. A new version of this Application has been designed to fit the method to the framework of the new CERN-PS control system through the concept of man-machine interface and using workstations[3].

This method is based on the measurement of beam profiles at three different secondary emission monitors with known transfer matrices between them. The three monitors are located downstream of the PS injection region, each of them measuring a horizontal and a vertical profile.

With w_i the half beam widths or heights measured at the i-th monitor position and $\beta_{i\,0}$ the β-functions of the machine (i.e. of the acceptance ellipse), the beam emittance and the geometrical emittance increase due to mismatch of the incoming beam[4] may be written as

$$\varepsilon = \underline{G}_i \frac{w_i^2}{\beta_{i\,0}} \qquad (7)$$

and

$$\frac{\Delta\varepsilon}{\varepsilon} = |k| \left(\frac{|k|}{2} + \sqrt{1 + \frac{|k|^2}{4}} \right) \qquad (8)$$

with

$$k = \frac{1}{\sqrt{\underline{G}_i}} \left((\underline{G}_i - 1)^2 + j\, \underline{B}_i^2 \right) \qquad (9)$$

where j is the imaginary unit, $\underline{B}_i$ and $\underline{G}_i$ are the normalized matching parameters which describe the beam emittance ellipse and k is the complex mismatch vector of the beam[2].

The mismatch vector k provides a measure for the comparison of the emittance ellipse of the incoming beam with the acceptance ellipse of the PS in the injection region. The modulus of k has to be much less than unity for good matching. For instance, |k| is equal to $1/\sqrt{2}$ when the emittance blow up reaches 100%.

From the three half beam sizes w_i measured with monitors at different positions and with the knowledge of the transfer matrices between these monitors, the normalized matching parameters at one monitor position, say $\underline{B}_1$ and $\underline{G}_1$, may be expressed as functions of the w_i, $\beta_{i\,0}$ and the phase advance difference $\Delta\mu_{i\,j}$ between monitors i and j[5].

When the beam is perfectly matched, the normalized matching parameters take the values $\underline{B}_1 = 0$, $\underline{G}_1 = 1$. Hence the beam emittance may be estimated from a unique half beam size measured with a monitor, say w_1, using Eq. (7) in which $\underline{G}_1$ is set to unity.

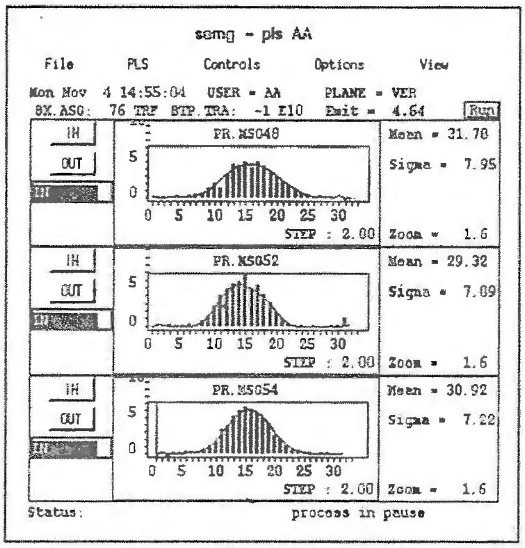
A convenient definition of the half beam sizes when the beam profile density is not exactly known consists in taking for w_i twice the r.m.s. value of the profile distribution.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S18BPA03

In the presence of dispersion, the pure betatron half beam width $w_{\beta\,i}$ must be used instead of the overall half beam width w_i which includes both the betatron and momentum dispersion contributions. No such correction has been implemented until now.

All acquired profiles are analyzed prior to the beam size evaluation since faulty grids, fluctuations in grid output signals and random disturbances on the beam may perturb the measurements. The treatment of the profiles are performed only on the user request, and consists of the following:

- elimination of faulty grid measurements,
- treatment of the tails (for base line detection),
- profile smoothing using spline functions.



Mon Nov 4 14:55:04 USER = AA PLANE = VER
BX.ASG: 76 TRF BTP.TRA: -1 E10 Emit = 4.64

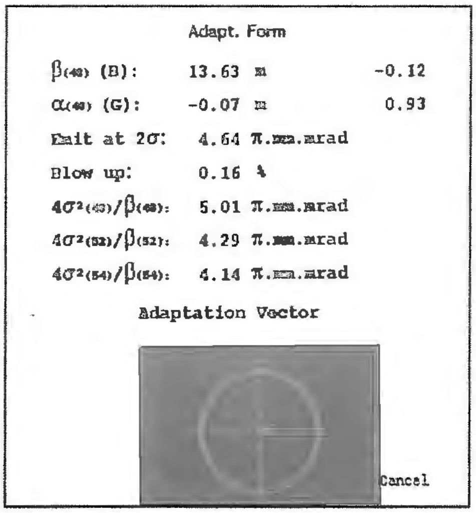
PR.MS048 Mean = 31.78 Sigma = 7.95 STEP : 2.00 Zoom = 1.6

PR.MS052 Mean = 29.32 Sigma = 7.09 STEP : 2.00 Zoom = 1.6

PR.MS054 Mean = 30.92 Sigma = 7.22 STEP : 2.00 Zoom = 1.6

Status: process in pause

Fig. 2: Display of transverse beam profile distributions at three different monitors. The bars charts show beam profiles measured with 32 wire grid monitors. The continuous lines show the smoothed profiles after elimination of erroneous data (the smoothing coefficients are chosen by the user). Beam emittance, mean and r.m.s. values of the smoothed profile distributions are given.



$\beta_{(48)}$ (B): 13.63 m -0.12
$\alpha_{(48)}$ (G): -0.07 m 0.93
Emit at 2σ: 4.64 π.mm.mrad
Blow up: 0.16 %
$4\sigma^2_{(48)}/\beta_{(48)}$: 5.01 π.mm.mrad
$4\sigma^2_{(52)}/\beta_{(52)}$: 4.29 π.mm.mrad
$4\sigma^2_{(54)}/\beta_{(54)}$: 4.14 π.mm.mrad

Fig. 3: Display of the mismatch vector in polar form. The area within the circle fits to emittance blow-up below 100%. The 2-r.m.s. emittance, the emittance blow-up due to mismatch, and the normalized beam matching parameters ($\underline{B}_1, \underline{G}_1$) at the 1st monitor location, with their corresponding Twiss parameters (α_1, β_1), are shown. Estimations of the beam emittance obtained from half beam sizes measured with the three monitors, regardless to beam matching, are shown for comparison.

IV INJECTION TUNING

A method for the correction of oscillations at 1 GeV injection into the PS was introduced in a early version of the control system[6]. As in the case of the emittance and matching measurement process, an up-to-date version of the hadron injection process has been assembled in order to be integrated in the new CERN-PS control system.

The amplitude and the phase of the pure betatron injection oscillation when the closed orbit is unknown are derived by adjusting a sine curve to the normalized beam trajectory difference between two consecutive turns measured shortly after injection. Moreover when the machine tune is also not known, the optimum tune value which minimizes the summed-square error between the fitted sine curve and the normalized measured trajectory difference is determined by iteration. This yields a satisfactory knowledge of the betatron oscillation at injection. Hence the correcting strengths to be applied to a given pair of injection steering elements in order to cancel this oscillation are evaluated accordingly.

For the beam trajectory measurements, 40 pick-up monitors are installed around the PS ring, each measuring a horizontal and a vertical beam position with respect to the pick-up center. Trajectory position at monitor i in machine turn number n can be written as

$$x_{i\,n} = z_i + A\sqrt{\frac{\beta_i}{\beta_R}}\cos(\mu_i + \phi + 2(n-1)\pi Q) \qquad (10)$$

in which z_i is the closed orbit value at the i-th monitor location, μ_i, β_i are the phase advance and the β-function at this monitor position relative to a reference location R in the machine, Q is the tune (not an integer value), and A, ϕ are the amplitude and phase of the betatron oscillation respectively.

The normalized trajectory difference between two successive turns is then independent of the closed orbit and may be written as

$$\sqrt{\frac{\beta_R}{\beta_i}}\,\frac{x_{i\,n} - x_{i\,n+1}}{2\sin\pi Q} = C\sin\mu_i + D\cos\mu_i \qquad (11)$$

where C and D depend on A, ϕ, Q and the turn number n.

Least square approximation of measured beam trajectory difference provides estimators $\underline{C}$ and $\underline{D}$ of variables C and D by minimizing the summed-square error

$$\sigma^2 = \frac{1}{N-2}\sum_{i=1}^{N}\left[\sqrt{\frac{\beta_R}{\beta_i}}\,\frac{x^m_{i\,n} - x^m_{i\,n+1}}{2\sin\pi Q} - \underline{C}\sin\mu_i - \underline{D}\cos\mu_i\right]^2 \qquad (12)$$

where $x^m_{i\,n}$, $x^m_{i\,n+1}$ are the measured beam trajectory in two consecutive turns, N is the number of monitors providing reliable signal data.

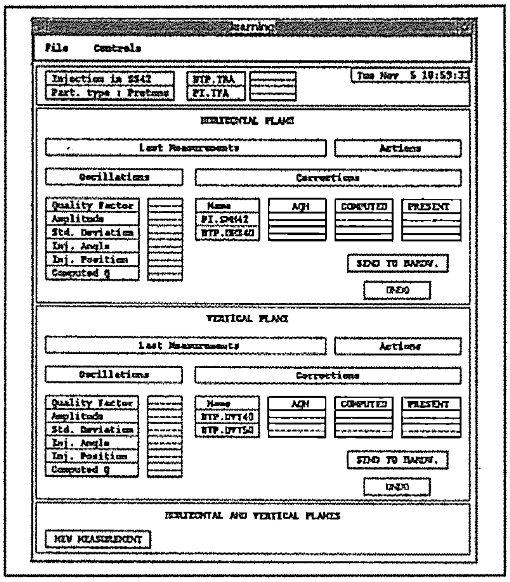
In case of faulty monitors, the rejection of erroneous acquisitions is based on usual statistical criteria to decide whether the chance of occurrence of some particular measured beam position is less than some fixed number.

Hence approximate values $\underline{A}$ and $\underline{\phi}$ of the betatron oscillation amplitude and phase can be derived from the estimators $\underline{C}$ and $\underline{D}$. The position and angle variations Δx_0, $\Delta x'_0$ of the beam at injection point with reference to the closed orbit can then be obtained from Eq. (10). Finally the current variations values ΔI_1, ΔI_2 to be applied at the two correctors (septum and steering dipole) to cancel the injection oscillation are derived from these position and angle by a linear transformation written as a 2 by 2 matrix equation. The components of the transformation matrix can be either determined experimentally or from a programme for lattice design.

The summed-square error σ^2 near the real machine tune Q is well approximated by a parabola. Therefore a search procedure may be carried out to estimate the unknown tune value Q. The Fibonacci search has been considered as an optimum Q-seeking method. This procedure starts from the initial search interval, and successively reduces the subsequent search intervals by means of a sequence of numbers, which are determined from the Fibonacci numbers. The number of search steps is fixed by advance. The Fibonacci search is the most effective one-dimensional search strategy available. An accuracy of 0.01 is then achieved with 8 search steps within an initial tune search interval of 0.5. However, in some cases fluctuations in the monitor acquisitions may be distributed in such a unpredictable way that the tune calculated in this way is irrelevant. Assuming that the trajectory measurements are done in the 1st and 2nd turns, the tune error is inversely proportional to the signal to noise ratio. For instance, rough estimate shows that when the amplitude of the oscillation at injection is lower than 5 mm, the tune accuracy cannot be kept within 0.04.

Consequently the tune search is mostly reliable at the first stages of the correction process, when the injection oscillations still have large amplitudes. Once an adequate estimation of the tune has been found during the first few corrections, the latter Q-value can be frozen and further corrections may be carried on to refine the injection tuning. Simulation from the trajectory data has shown the validity of the above algorithm.

At any stages of the tuning process, no automatic corrections will be performed, the computed corrections are merely proposed to the user, who decides whether they have to be carried out. Fig. 4 shows the proposed Workstation window for the horizontal and vertical interactive corrections of betatron oscillations at 1 GeV injection into the PS.



Injection in SS42
Part. type : Protons

Fig. 4: Display of the interactive window used for the injection tuning. For every two single turn trajectory measurement, the computed tune, the fit quality and the characteristics of the derived betatron sinewave are anticipated. The appropriate currents to be applied to each pair of correctors will be displayed for possible user action.

V CONCLUSION

The injection timing control and the emittance and matching measurements are presently implemented within the new CERN-PS control system and are used in current operation to the user satisfaction. The third application, optimization of the betatron oscillations at injection into the PS, although well under way, is not entirely set up. The search of the optimum tune and the calculation of the two corrections which will cancel the oscillations are finalized, but the final stage, on-line connection to the beam trajectory measurements, is in the process of being implemented.

VI REFERENCES

[1] J.P. Riunaud, "Modelling of a Timing Process", Internal Report CERN PS/PA Note 91-21, November 1991.

[2] M. Martini, "Calcul de la désadaptation dans la ligne d'injection à 800 MeV du PS", Internal Report CERN/PS 84-12 (OP), August 1984.

[3] M. Boutheon, F. Di Maio, A. Pace, "General Man-Machine Interface used in Accelerator Controls: Some Applications in CERN-PS Control Systems Rejuvenation", these Proceedings (ICALEPS, Tsukuba, Japan Nov. 1991).

[4] K. Schindl, "Increase of 95% Emittance due to Missteering and Mismatch", Internal Report CERN MPS/BR Note/75-25, November 1975.

[5] P. Brummer, "The Method of Measurement of the Emittance and the Betatron Phase Space Parameters in the Beam Transfer System of the ISR", Internal Report CERN-ISR-OP/72-6, January 1972.

[6] M. Martini, J.P. Potier and T. Risselada, "Automatic Injection Tuning Using Two Successive Single Turn Trajectory Measurements", in European Particle Accelerator Conference EPAC 90", Nice, France, June 1990.

CARSO - A Program For Automated First Turn Steering

Mark Plesko*
Sincrotrone Trieste
Padriciano 99, I-34012 TRIESTE
e-mail: plesko@elettra-ts.infn.it

Abstract

CARSO is a program package which contains several new software tools to be used during first turn steering of a storage ring, or during the steering through a beam transfer line. CARSO includes routines which check the effects of magnetic components on the beam, check the measurements of the beam position monitors and simultaneously steer the beam through the ring to perform one full turn. The programs are written in ANSI extended standard FORTRAN 77 and comprise 6000 lines of source code, 87 subroutines and about 1000 different variables. The concepts used within CARSO are presented.

I. INTRODUCTION

The conventional approach to first turn steering is first to adjust manually some steering elements so that the beam reaches the last beam position monitor, and then to apply numerical orbit correction procedures to reduce the r.m.s of the orbit displacements measured at the beam position monitors [1,2]. During the manual phase the machine physicists often have to make some checks and reason about possible malfunctions of the equipment such as incorrect beam position measurements, quadrupole polarity errors, ineffective steerers, etc.

CARSO automizes the first phase and combines it effectively with the second. CARSO performs routine equipment checks systematically, notifies the operator when some possible errors have been found and tries to correct for some of them at their source. For the ones it can not correct, CARSO provides a fine adjustment of the beam towards the beam axis in a semiempirical way.

CARSO combines for the first time two concepts which itself are rather new for machine physics software. First, it actively uses the beam itself as a measuring device to test the components of the ring (a possibly similar approach is described in [3]). Before steering, first the relevant components are checked by sweeping the beam. Only if they behave as expected, they are used to adjust the beam towards the beam axis. The second concept is that rather than comparing the model of the storage ring with the measured beam positions globally, CARSO checks each monitor and the elements surrounding it separately (the same philosophy is applied in [4]). Thus it is able to localize gross errors rather precisely.

CARSO is capable of steering the beam successfully in spite of certain types of component malfunctioning, because it can recognize a few classes of error patterns and correct its model of the ring accordingly. CARSO still provides many interactive input/output messages and prompts, in order to give the human operator complete control over its performance. In this respect, CARSO can be viewed as an "apprentice" doing the simple and obvious routine tasks, reducing the load of the operator, who can thus concentrate on global reasoning.

CARSO uses machine theory as well as some heuristic experience of a skilled operator to evaluate the checks. If a problem is encountered that can not be solved by using this knowledge, CARSO informs the operator and waits for his/her intervention. Sometimes CARSO may "overlook" a severe error, but from its output the operator is nevertheless able to detect the error or to get at least some hints about the problem.

II. THE STRUCTURE OF CARSO

A. The Guidelines

The guidelines for the development of CARSO were:

- compare the predictions of the theoretical optics, which is represented by the model, with the measured beam position ;
- in case of a mismatch between the two, correct either directly the source of the error, correct the model, or at least minimize the effect of the error on the beam trajectory;
- make as little as possible assumptions, that can not be verified by a measurement - i.e. use the model only for comparison and not for calculations;
- avoid pure quantitative checks if qualitative ones can give reasonable answers, because there might be too many unexpected errors, degrading the numeric precision below acceptable levels;
- ensure that no action taken by CARSO sets the hardware beyond its limits;
- adjust algorithms to include heuristic rules of a human operator.

B. The Approach Chosen

For the purpose of first turn steering, when the beam traverses each element only once, a storage ring can be viewed as a simple beam transfer line. In a transfer line the basic approach is to adjust the elements so that the beam travels from one monitor to the next and so on, until the last monitor is reached. If the beam does not reach a monitor, it is obvious

---

* Supported by an ICTP fellowship.

that the error must be with an element that is located upstream of the monitor, most probably between the last reached and the first missed monitor.

It is therefore natural to divide the beam line in groups of elements between two adjacent monitors. CARSO steers the beam from one group to the next and stops when the last monitor is reached.

One such group is named a 'monitor group'. The whole structure of CARSO is built to first check one group and then to fine adjust the beam so that the BPM readings of the two monitors forming the group boundaries are close to zero. CARSO assumes that all upstream groups are without errors, because they have already been checked. Also the upstream monitor of the group is assumed to be O.K., since it has been checked when it was the downstream monitor of the previous group.

The assumption that all the elements upstream from the monitor group are errorless is essential. It allows CARSO to use all upstream steerers and monitors to localize a possible error within the monitor group. To be precise, CARSO assumes only that the upstream dipole and quadrupole strengths are correct. It is not affected by transverse positioning errors, because it performs difference measurements. Even if the steerer that is used for the difference measurement is not correct, CARSO will spot it in most of the cases, because it checks the monitor group by measuring the beam position in the two monitors. The actual steerer strengths do not enter into the calculations.

If some elements between the steerer and the upstream monitor of the monitor group are incorrect, and they have been overlooked during checks of upstream groups, CARSO might assume a steerer error or an error in the monitor group. CARSO does not have the complexity to spot such effects. Therefore CARSO provides sufficiently rich output messages, which allow the operator to decide, whether there was really an error in the monitor group that was under investigation. CARSO changes its model of the ring to comply to its measurements only if the operator confirms the suggested change.

Even if the operator's decision is wrong, CARSO will soon recover from erroneous assumptions, because once the faulty elements are upstream of the steerer which is used to check a monitor group, they do not influence the measurements any more.

The only limitation on the performance of CARSO comes from the measuring precision of the beam position monitors. CARSO uses the measured beam position at upstream monitors to extrapolate it (by means of the model) to the downstream monitor of the monitor group and to compare it with the measured beam position. If the measuring errors are too large, they screen all other errors that come from a mismatch of the real lattice with the model. The limiting measuring precision depends on the lattice chosen and on the monitor spacing.

C. The Errors Looked For

As the beam position measurement during first turn steering is never very accurate, CARSO looks only for large element errors, such as:

- large beam energy errors or dipole field errors, respectively;
- quadrupole polarity errors;
- BPM button electrodes cabling errors;
- "hot" or dead button electrodes or whole monitors, respectively;
- lumped steerer cabling errors (wrong polarity or exchanged planes);
- incorrect steerer current/strength relations.

By comparing the predictions from the model with the measurements, CARSO is also able to detect larger deviations from the nominal element strengths and BPM imperfections, but it is not able to localize them uniquely, unless there is only one element between two monitors. This is left to the human operator who has much more ability for global reasoning than any software.

D. The Actions Taken

If CARSO encounters problems, it either corrects the errors at the source or tries to overcome the effects of the errors. CARSO can perform the following actions:

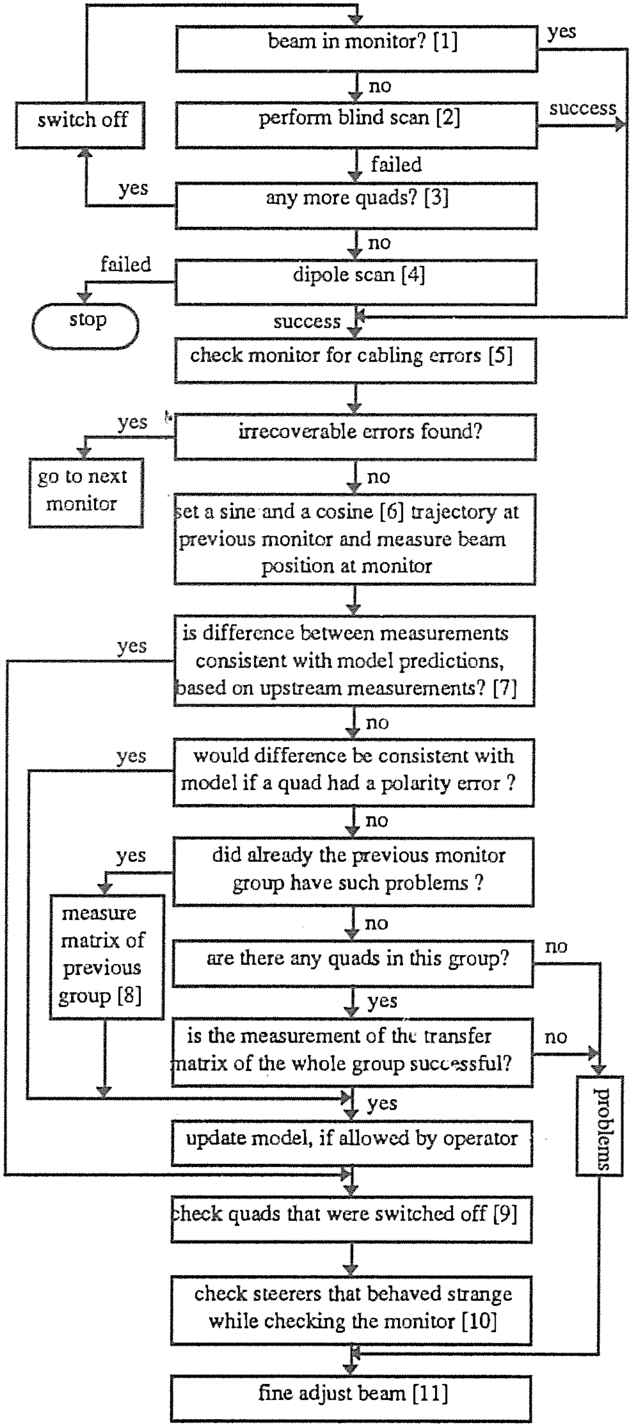
- recalculate the beam position from BPM button signals, if buttons are miscabled;
- reassign steerer polarity and plane, if the steerer's power supply connections are wrong;
- change the sign of a quadrupole strength in the model, if the quadrupole has a wrong polarity - CARSO is then able to continue and still complete the first turn, although the operator would usually stop the processing in such a case;
- measure the transfer matrix of a whole group of consecutive elements and place this matrix into the model for continuation of CARSO, if the error can not be localized to one single element;
- look at downstream monitors, if the monitor shows no beam;
- perform a systematic two dimensional blind scan using one or several steerers simultaneously, if the beam is lost or does not reach the next monitor;
- adjust the beam position to zero at two consecutive monitors with one or two steerers, in both planes, if the beam position reading is outside the tolerance given by the BPM resolution;
- scan the current of the dipole around the nominal value, if the monitors which are downstream show no beam even after a blind scan;
- switch off a quadrupole and measure its current/strength relation.

The last two actions are seldom performed because it usually takes a lot of time to change the power supply currents of dipoles and quadrupoles by large amounts.

If CARSO encounters steerer and monitor errors that it can not correct, it simply discards the steerer or monitor, respectively, and does not use them any more during its run.

E. The Flowchart of CARSO

For each monitor, CARSO performs the following algorithm:



The notes mean:

[1] If there is no beam in the monitor, downstream monitors are also inspected. If one of them gives a positive signal, then the monitor group is extended up to this monitor and all the monitors in between are marked to be dead.

[2] The blind scan is first done with the most effective steerer, then with the next effective steerer and so on, until it succeeds to get a signal from any downstream monitor. If it does not succeed with single steerers, it uses several in parallel.

[3] CARSO takes into account that quadrupoles that are powered in series with upstream quadrupoles can not be switched off.

[4] Only one dipole in the group is scanned, since usually the dipoles are powered in series. Also the energy error is either spotted already in the first dipole, or never.

[5] The monitor is checked by sweeping the beam across all its four quadrants. Up to three steerers are used, to rule out possible steerer errors. Therefore also wrong steerers are detected in this routine.

[6] A sine trajectory means that the beam position at the monitor is zero and its slope is maximal. A cosine trajectory has zero slope and a large displacement from the beam axis. First, a cosine trajectory in x and a sine trajectory in z are established, then a sine trajectory in x and a cosine trajectory in z. If either one of these can not be set due to limited steerer strengths, only a simple difference measurement is performed.

[7] The beam position in the second upstream monitors must be measured to determine the slope of the beam in the first upstream monitor. Here the model of the ring between the two monitors must be used. CARSO assumes that it is correct, because it has been checked in the previous monitor group. From the determined position and slope at the upstream monitor, the expected beam position at the checked monitor is obtained by matrix multiplication.

[8] If the previous group was already wrong, then the check is meaningless, because the slope of the beam at the upstream monitor of the monitor group can not be determined correctly (see [7]). However, if the current monitor group is correct, then at least the transfer matrix of the whole previous group can be measured and stored, so that the error does not influence calculations of transfer matrices of elements that are further upstream. If there is a steerer within this previous group, it is marked to be locked, set to zero and not used anymore.

[9] The current in the quadrupole is increased stepwise and for each step the quadrupole strength k is measured in both planes and compared to the expected one.

[10] The steerers that could not move the beam significantly or looked as if their cables are misconnected, are checked numerically by comparing their measured M_{12} transfer matrix element with the one calculated from the model.

[11] The beam is adjusted such that the measured position at the upstream and at the downstream monitor of the

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S18BPA04

monitor group is minimal, i.e. below the measuring imprecision of the monitors. This is achieved in both planes separately with one or two steerers. CARSO provides two options - either the use of measured M_{12} transfer matrix elements for the steerers or the use of the ones from the model.

III. THE CURRENT IMPLEMENTATION AND ITS LIMITATIONS

CARSO is written in ANSI extended standard FORTRAN 77 and has been successfully compiled and run under VAX/VMS and HP-UNIX operating systems. It has been tested and studied with a linear simulation of the storage ring ELETTRA [5], which simulates transverse magnetic element and BPM positioning errors, dipole quadrupole and steerer strength errors, quadrupole and steerer polarity errors, and BPM button electrodes cabling errors.

The model which is used by CARSO is a linear thick lense model using transport matrices, without skew elements. Fringing fields and nonlinear fields are not taken into account, neither is the finite beam size nor the BPM nonlinear response. However, the BPM nonlinear response which is due to the field distortion of the vacuum chamber wall, can in principle be corrected for by the BPM software.

Besides the procedures to check and steer one monitor group, as described in the previous chapter, CARSO also includes an initialization and an epilogue. The initialization performs the following:

- initalizes the CARSO variables and defines output screens;
- reads the data describing the model of the ring from a file;
- calculates the transfer matrices of each element;
- defines a list of effective steerers for each monitor;
- measures the launch (injection) coordinates (x,x',z,z').

The launch coordinates can be measured only if there are at least two monitors between the injection point and the first bending magnet and if the quadrupoles in between are switched off. This is feasible for ELETTRA [6], but may not be so for other rings, in which case the launch conditions must be supplied by the operator.

The epilogue just displays some statistics about the beam position in each monitor, the r.m.s. of the measured orbit and the maximum steerer strengths.

CARSO was built with the electron storage ring ELETTRA in mind. Although CARSO is a general program, which can take any lattice from an input file, some details are nevertheless specific for a ring like ELETTRA.

It is obvious that - since CARSO runs automatically - nondestructive beam position monitors must be used. Fluorescent screens could be adopted only if they were moved into the beam pipe automatically and providing the beam position could be read out by software.

The check of the BPM button electrodes cabling errors works only for BPMs with four button electrodes, where the buttons are placed around the centres of the respective quadrants. If the buttons are placed on the x and z axes, the subroutine which checks the BPMs must be changed or skipped.

The next site specific algorithm is due to lumped steerer magnets. The current implementation of CARSO supports only combined horizontal/vertical steerer magnets. CARSO generally treats the two planes separately but for two cases: the blind scan and the monitor check are done with both planes of one steerer. Hence the most effective steerer is defined to be effective in both planes. For rings with single plane steerer magnets, the data structure also will have to be changed.

A minor change in the initialization of CARSO will have to be made if pure dipole magnets or sector magnets are in the lattice. ELETTRA has only hard edge dipoles with a nonzero index. However in this case only the subroutine defining the transfer matrix of a dipole must be slightly changed as well as the input statements to distinguish between different types of bending magnets.

IV. CONCLUSIONS

CARSO replaces successfully the machine operator for routine tasks during first turn steering, once the hardware is stable and reproducibly controlled by the control system. However, it requires a reasonable beam position monitor spacing and measuring precision and full computer control over the equipment.

It is planned to use CARSO during the commissioning of ELETTRA and also after longer periodical shut downs. Being tailored to the specific ELETTRA components, CARSO can also be used for first turn steering of other storage rings with little changes to the software.

V. ACKNOWLEDGEMENTS

I am happy to thank Albin Wrulich for the initiation of the original idea and for many other discussions concerning this topic. Comments and suggestions on the algorithms were given also by Carlo Bocchetta and Ryutaro Nagaoka. Antonio Choi thoroughly debugged CARSO and helped me to find many errors. Many thanks to them all.

VI. REFERENCES

[1] J. Milutinovic and A.G. Ruggiero, "First Turn Around Strategy for RHIC", Proc. PAC 1991, San Francisco, 1991.

[2] A whole list of closed orbit correction procedures may be found in D. Brandt, W. Herr, J. Miles and R. Schmidt, "A New Closed Orbit Correction Procedure For the CERN SPS and LEP", Proc. ICALEPS 89, Vancouver, 1989, or in J. Sheppard et al., "Beam Steering in the SLC Linac", Proc. PAC 85, Vancouver, p.2180, 1985.

[3] D.L. Lager et al., "MAESTRO A Model and Expert System Tuning Resource for Operators", Proc. ICALEPS 89, Vancouver, 1989.

[4] M. Lee, S. Kleban, "SLC Bean Line Error Analysis Using A Model Based Expert System", Proc. EPAC 88, Rome, 1988.

[5] ELETTRA - Conceptual Design Report, Sincrotrone Trieste, Italy, April 1989.

[6] M. Bassetti, C.J. Bocchetta, A. Wrulich, "Magnet Lattice for the Sincrotrone Trieste", ST/M-87-11, August 1987.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S19SP01

Control Protocol : The Proposed New CERN Standard Access Procedure to Accelerator Equipment. Status Report

G. Baribaud, I. Barnett, G. Benincasa, O. Berrig, R. Brun, P. Burla, A. Burns, R. Cappi,
G. Coudert, C. Dehavay, B. Desforges, R. Gavaggio, G. Gelato, H.K. Kuhn, J. Pett, R. Pittin,
J.P. Royer, E. Schulte, C. Steinbach, P. Strubin, D. Swoboda, N. Trofimov (*), L. Vos
CERN European Organisation for Nuclear Research, 1211 Geneva 23, Switzerland
(*)Home Institute IHEP Institute for High Energy Physics 142284 Protvino, USSR

Abstract

Control protocol provides a normalized access procedure for equipment of the same kind from a control system. Modelisation and the subsequent identification of functionalities with their parameters, variables and attributes have now been carried out at CERN for representative families of devices.

ISO specifications, such as the ASN.1 metalanguage for data structure representation and MMS definitions and services have, to some extent, been introduced in the design for generality and compatibility with external world. The final product of this design is totally independent of the control systems and permits object oriented implementations in any controls frame. The present paper describes the different phases of the project with a short overview of the various implementations under development at CERN.

I. INTRODUCTION

Studies on protocols have been carried out at CERN for more than three years. The basic ideas have been set up in the frame of the Technical Board for Controls and Electronics (TEBOCO): this consultative Board had the mandate of investigating and proposing uniformisation and standardization in the concerned field.

The generalities of the control protocol and the results obtained with first prototypes implementations, have been presented at the Accelerator Control Conference in Vancouver, October 1989 [1] [2].

At the beginning of 1990, a working group called WOPRO (Working Group for Protocols, whose members are the Authors of this paper) was set up with the CERN Management mandate of studying and proposing control protocols for all accelerators at CERN.

Studies on Protocols have been carried out by WOPRO through two activities :

i) the different CERN equipments have been grouped in classes of similar devices. For each class, behavioural models have been proposed and the corresponding functionalities with the associated parameters have been identified. Appropriate structures for representing data have also been proposed. This activity, which is independent of the control system layout, has been carried out by the specialists of the WOPRO group.

II) the Control Protocol must be implemented in the actual CERN controls structures. This activity concerns more precisely all those services allowing the external visibility of the protocol, i.e.the access procedures to the equipment, and the software structures required by the protocol realisation. This implementation study is carried out by controls specialists together with the WOPRO members.

The first and main activity of WOPRO (conceptual design phase) has been terminated by mid 1991 [3] [4] [5] [6]. The second, implementation oriented phase, is under study and major results are expected for spring 1992.

II. MAIN CHARACTERISTICS OF THE CONTROL PROTOCOL

Standardization and uniformisation of equipment access is not a novelty in accelerator controls field. In fact the control system of each accelerator or Complex has introduced its own standard. What is different in the proposed WOPRO's approach, can be summarized in the following five points :

- The investigations have been carried out CERN wide. Each considered class of devices includes examples coming from the more concerned machines.

- The study has at first been bottom-up oriented, from the equipment to the control system. The proposed protocols fulfills then principally the needs of the users of the control systems at CERN.

- The functional description of the devices includes all aspects related to operation. In the accelerator field a device works very often in close connection with other equipment that is necessary for the accomplishment of its activity (triggering systems, function generators, etc.). The proposed protocols consider these equipment as part of the device and include them in the design.

- The design is based on behavioural models. For each family of the considered devices, the relevant specialists have firstly developed one or several behavioural models: the model includes all aspects that are necessary for an operational use of the device. This conceptualization has provided the degree of abstraction needed for the generality of the design.

- An object oriented approach has been used. The user has to specify "what" to do: the object-device knows "how" to do it. This allows a large independence between the implementations of the controls specialists and those of the device specialists. Other features of the object oriented design, such as class structures, inheritance, etc., are proposed in the implementation phase.

III. EXPECTED ADVANTAGES OF USING CONTROL PROTOCOLS

One can summarize the positive aspects of control protocols as follows :

- seen from the top (operation), they provide a uniform visibility to the application programs for a class of devices.
- seen from the bottom, the device specialist can totally dedicate his skill to solving his specific problems in the more suitable way.
- as a consequence of the previous point, control protocols promote a better use of competences. Where the controls activities are mixed with the activities of the specific equipment, each party has to learn implementation details of the other party, that are not of his domain of competences. Obviously, this inconvenience is largely reduced with control protocols.
- a clean separation of responsibilities is introduced between controls and equipment specialists. The two independent realizations that communicate with each other only by exchanging well defined messages, permit to fix the position of this ideal "red line" in a natural way at the level of the messages themselves. This is particularly important during fault finding on a system, where the problem of determining which specialist is concerned arises.
- as a consequence, the maintenance of the systems is simplified and facilitated. We recall that the cost in man-power for the maintenance is estimated at about two thirds of the total cost, calculated on the total lifetime of the device.

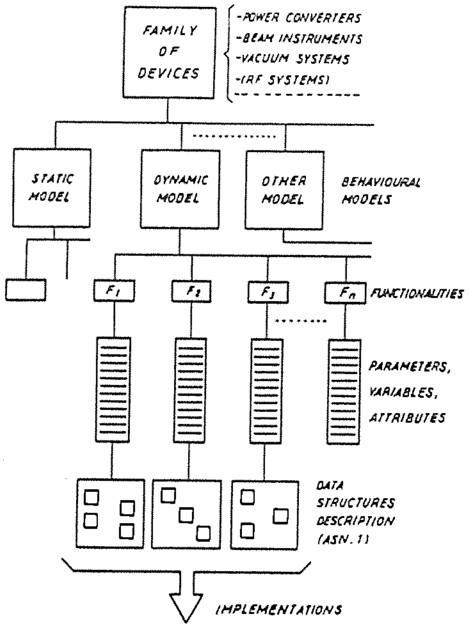


Figure 1. Design phases of control protocol

IV. CONCEPTUAL DESIGN OF CONTROL PROTOCOLS

As already mentioned in the Introduction, the design phase of the control Protocols has been divided in a sequence of interdependent activities: the output (conclusions) of each activity is used as input (assumptions) for the next one (Figure 1). In order, these activities have been :

A. Definition of a Class (Family) of Devices

All those devices with similar characteristics and (or) similar goals, belong to the same class. The first three big classes of devices that we have considered are :

- The Power Convers
- The Beam Instrumentation devices
- The Vacuum systems

B. The behavioural models

A model is an abstract representation of the behaviour of an object. In our case the object is a device symbolizing a class of similar devices as previously defined. The control protocols we propose are intended for an operational use of the devices and the specialists have limited their investigations inside this boundary. The result is a serie of models covering all the operational aspects of the concerned families of devices.

C. Identification of the Functionalities

In the behavioural models of a class of devices, all those activities that have a common goal in the operational sense can be grouped together to form a Functionality. As an example, five functionalities have been identified in the power converters model :

- Status_controller
- Settings_ actuator
- Measurements_actuator
- Trigger_sequencer
- Function_generator

D. Parameters of the Functionalities

Each single activity inside a given Functionality is accessible by the external world through an appropriate parameter. In general a parameter is composed of variables and attributes. Variables contain the setting values at a given instant. Attributes contain constants representing, in general, the limits of validity for the variables (max. and min. values, lists of discrete values, etc.).

For each Functionality, a complete list of parameters, variables and attributes, has been defined.

E. Data structures representation

With the identification and the definition of the parameters and their associated variables and attributes, one can consider that the first, important phase in the activity of the WOPRO is terminated. This design phase has produced a significant amount of inter-related data : an adequate tool of representing structured data should then be used. We have decided to use the Abstract Syntax Notation One (ASN.1) that is an ISO International Standard (ISO-IEC 8824) fulfilling our needs. ASN.1 uses a metalanguage that permits a simple data representation, totally independent of any specific computer environment : from that, the data structure can be easily translated into a specific source code.

V. GENERAL IMPLEMENTATION SCHEMES

As already mentioned, the details of implementation are being studied with the control Groups, using their standards and tools. The proposal described here is only a sort of block diagram, representing the basic entities necessary to implement the control protocol (Figure 2). The implementation is composed of two main parts, one specific to each device and one general for a class of devices. They exchange information through the standard defined messages and need data bases to keep relevant data. In more details :

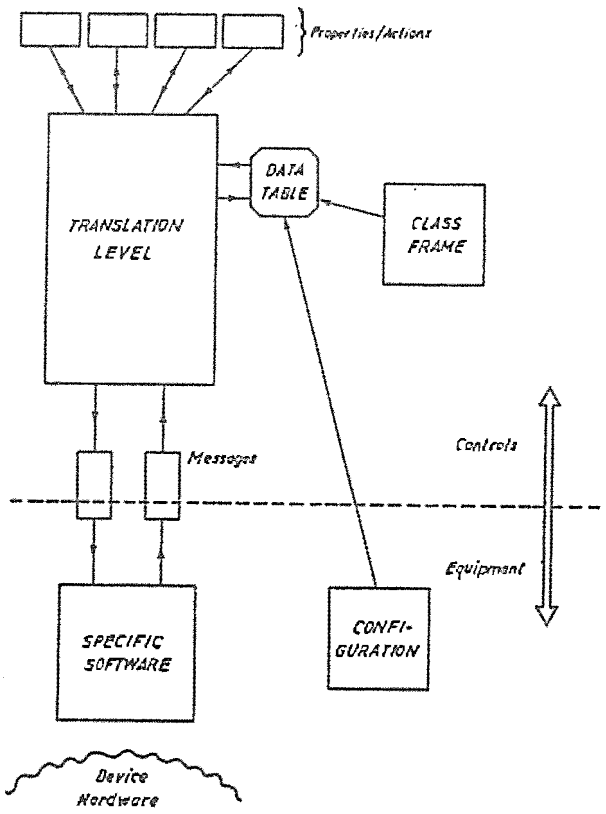


Figure 2. A general implementation scheme

A. Definition of Class Frame

The Frame contains a list and a structure definition of all visible functionalities, parameters, variables and attributes for a class of devices. In general the Frame does not contain values; the only exceptions are attributes: each attribute definition is accompanied by a complete, numbered list of all possible values in the process accessed by the control system. The frame should be housed in a central computer data base.

B. Definition of Configuration

The configuration contains the list and the values of all functionalities, parameters, variables and attributes for a single, specific device. The form and the structure of these entities are the same as for the class Frame. The configuration should be housed in the device itself and should be extracted using standard services during initialization procedure.

C. Definition of Messages

The messages contain the necessary information exchanged between control system and devices. They represent also the red line for separation of responsibilities. Their contents and their logical structure are independent of the controls architecture. Their physical structure depends on the controls features.

The message is composed of a certain number of fields corresponding, at the very most, to the number of Functionalities identified in the corresponding devices class Model. Each field contains a list of the parameters, variables and attributes characteristic for this Functionality, with their associated data.

D. Definition of Translation Level

This is a software module that translates the user requests into messages in protocol format. The Translation Level is that part of the control protocol that provides the access services to the equipment (see Introduction).It gives a uniform visibility of all equipment (within a class) to application programs. The implementation of this module strongly depends on the control system features.

VI. PROTOTYPES UNDER DEVELOPMENT

Three series of prototypes, using CAMAC technology, have already been developed in the PS Complex and are currently in operation. They concern the control of a dozen current beam transformers and their principles have already been reported [2]. These implementations have permitted unambiguous verification of some of the control protocols claimed advantages : in particular the large independence of the controls and specific developments and the software total cost reduction after the first implementation.

A new series of applications is now under development in the new CERN common control system [7]. This series

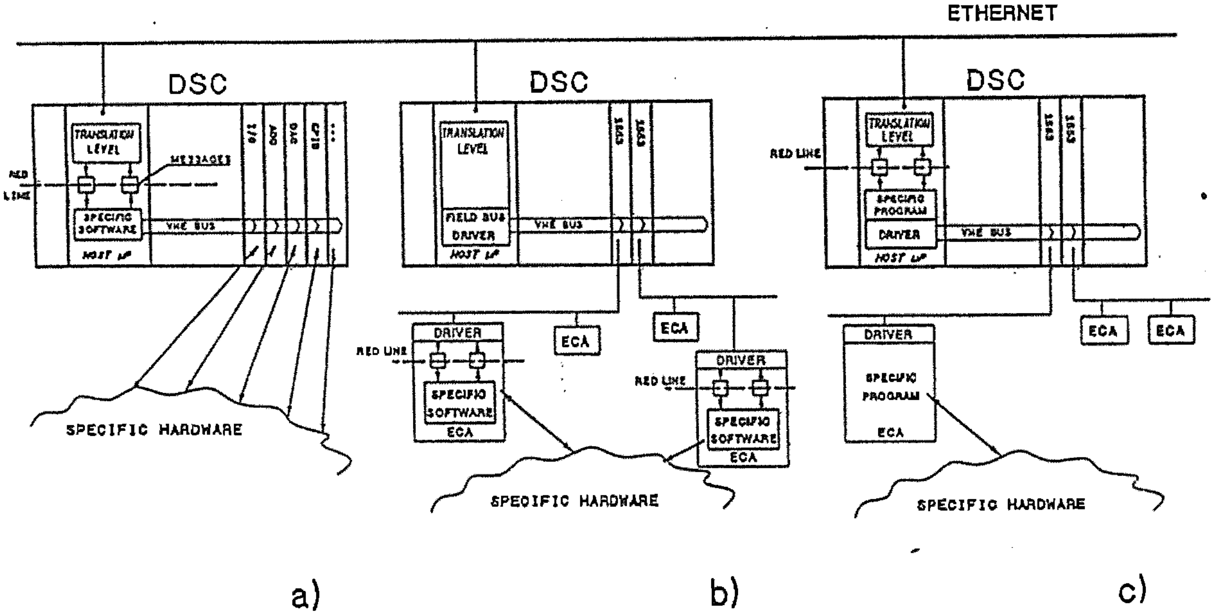


Figure 3. Actual implementation schemes

includes clusters of power converters and different types of beam instrumentation devices : the control protocol is implemented in the front end part of the new control system, called the DSC (Device Stub Controller) a VME crate using a 68030 as main processor and the LYNX RT-UNIX operating system.

Figure 3 presents three possible schemes adapted to different needs of the users, and that we briefly recall :

- Figure 3a represents the case of an equipment having all its I/O modules housed in the DSC : this configuration is more specially, but not exclusively, intended for beam instrumentation devices requiring intricate and complex data treatments. The specific and the control software share the same processor and exchange standard messages.
- In those cases where the equipment is composed of a cluster of similar devices attached to a field bus (power converters), the control protocol could be implemented as in Fig. 3b. The ECA's (Equipment Control Assembly) represent different types of specific crates standards as G64, VME, etc.
- Figure 3c represents a variant of 3b, where the different devices are not totally independent and require, still in the specific software, a common control action.

VII. REFERENCES

[1] R. Bailey, G. Baribaud et al. "Operational Protocol for controlling Accelerator Equipment" Int. Conf. on Accelerator Controls, Vancouver, October 89.

[2] G. Benincasa, L. Casalegno et al. "Implementation of a control protocol in the instrumentation field", Int. Conf. on Accelerator Controls, Vancouver, October 89.

[3] WOPRO members, reported by G. Benincasa, "Generalities on the operational control protocol", CERN-PS/CO Note 91-01.

[4] WOPRO members, reported by P. Burla, "Description formelle des paramètres et variables des convertisseurs de puissance pour la définition d'un protocole de contrôle", CERN SL-PC Note 91-86.

[5] WOPRO members, reported by P. Strubin, "Operational protocol for vacuum systems", CERN AT-VA Note 91-06.

[6] G. Baribaud, "Operational protocol for timing events", CERN SL-BI Note 90-05.

[7] The PS and SL Control Groups, "PS/SL controls consolidation project", CERN PS-CO Note 91-09 and SL-CO Note 91-12.

Man-Machine Interface Workshop Summary*

Stuart C. Schaller
MP-6, Mail Stop H-852
Los Alamos National Laboratory
Los Alamos, New Mexico, 87545, USA

Abstract

This report is a summary of the Man-Machine Interface Workshop that took place on 14 November 1991 as part of the 1991 International Conference on Accelerator and Large Experimental Physics Control Systems in Tsukuba, Japan. The conference was sponsored by KEK, the Japanese High Energy Physics Laboratory.

I. INTRODUCTION

The topic of man-machine (MMI) interfaces has received much attention in the general computing literature, (see, e.g., [1]). The Man-Machine Interface workshop at this conference was motivated by desires to provide an interactive forum for the discussion of how current methods and new ideas can be used to make communications between the accelerator control systems and the control system users as effective and comprehensible as possible. The goals of this workshop were two-fold:

1) To identify unifying principles in the design and implementation of man-machine interfaces for accelerator/physics control systems;

2) And, looking to the future, to encourage discussion of new, possibly speculative, man-machine interface techniques and their application to such control systems.

The 1989 ICALEPCS conference in Vancouver included a workshop on the use of workstations in accelerator control systems. Part of that workshop dealt with the problems and possibilities presented by the use of workstations as man-machine interfaces in accelerator control system environments. It is interesting to compare the changes in emphasis that have become evident in two years. Comments from the 1989 workshop emphasized several aspects of the man-machine interface including: the need for realistic feedback for analog controls whether implemented through physics knobs or a window interface; the desirability of multiple, non-overlapping screens on each operator console; and the importance of tools for rapid prototyping.

As part of the 1991 MMI workshop, we hoped to encourage discussion of a wide variety of topics, including: display methodologies, interaction techniques, human- and software-related engineering concerns, user construction of interfaces, and speculative aspects of three dimensional presentation and virtual reality. In this year's session, we were particularly interested in learning about the current use of windowing systems and third party interface builders, and the construction and maintenance of interfaces by users.

II. PAPERS

The Man-Machine Interface Workshop at this conference featured four invited papers and a discussion period. The papers were selected based on the abstracts submitted to the conference program committee. The full papers are available elsewhere in these proceedings.

Kevin Cahill described the uses Fermilab has made of the X-Window environment for accelerator control consoles [2]. Fermilab is using DEC Vaxstations. A single keyboard/trackball and set of knobs is interfaced to multiple screens using a locally engineered interface box. The X-Windows environment has been exploited to allow remote consoles across long haul networks and to support multiple consoles on a single workstation. A Fermilab console was actually running on a Vaxstation at KEK during the conference. Read-only consoles and consoles with limited command capability help to allay feelings of unease among their operators. In addition, all commands are logged on a central server.

Frank Di Maio from CERN discussed the workstations that are being introduced as part of the rejuvenation of the CERN control systems [3]. The rejuvenation effort is based on Unix workstations with X-Windows, Motif, and TCP/IP communications. CERN's first attempts included console emulation for some of the NODAL-based applications. The workstation environment includes a user interface editor and an interactive application builder.

A completely new system based on Unix, X-11 and the PHIGS graphics standard was described by Franco Potepan of the ELETTRA Synchrotron Light Source in Trieste, Italy [4]. The ELETTRA system has a very natural interface that allows direct access and manipulation based on CAD-derived pictures of the entire accelerator complex. This interface was an attempt to continue the desktop metaphor in the accelerator environment.

---

* Work supported in part by the U.S. Department of Energy

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S20PD01

Subrata Dasgupta of the Variable Energy Cyclotron Centre in Calcutta, India, talked about a method of portraying all four dimensions (x, x', y, and y') of the transverse beam phase space on a single two-dimensional screen [5]. His method used two-dimensional projections of suitably shaped three-dimensional solids. It did not provide as complete information as the usual ellipse representations, but did give a very intuitive feeling for what the beam was doing. He implemented these representations on an IBM PC.

III. DISCUSSION

In response to questions on control system security with proliferating consoles, Cahill said Fermilab handled such problems by limiting the capabilities of some consoles, by allowing the crew chiefs to observe all consoles, and by logging changes as they are made to the control system. Rusty Humphrey said that such security problems had never been raised as an operational issue at SLAC. Several other people indicated that control action logging was an effective way to determine cause and effect.

Questions regarding the use of Motif in interface construction were directed to Di Maio. He said they studied possible user interface management systems (UIMs) and decided that using the Motif interface was best. He noted that Motif was part of the environment already, and other products could be expensive and have an unpredictable market lifetime. Uli Raich commented that when using their knob widget the mouse served only to connect, the up and down arrow keys served to adjust.

In discussions related to how to present and interact with information via man-machine interfaces, Michael Crowley-Milling mentioned that consoles designed 5 to 6 years ago did not include keyboards, relying only on knobs, trackballs and touch panels. Since then, perhaps because of the widely accepted use of PCs and Macintoshs, keyboards have become an expected part of a console.

Regarding user construction of interfaces, Crowley-Milling pointed out that the NODAL interpreter at the CERN SPS resulted in the proliferation of displays. (In one example, a ship appeared on one SPS screen, sailed from screen to screen across the control room, and finally sank on the last screen.) Kevin Cahill said that with user expectations going up and the MMI environment becoming more complex, users themselves may not feel comfortable with building their own screens. Rusty Humphrey of SLAC noted that operators can go "berserk" in creating an extremely large number of approaches to screens. He said this trend is usually countered by senior operators who lose patience and get rid of some of the variety. The result can be a "relaxation oscillation" effect. Another conferee commented that he had success building prototypes in cooperation with operators, relying on programmers to put together the final production versions.

George Shering of CERN gave an informal summary of his view of the history of operator interfaces in accelerator controls. He said that beginning with the early LAMPF controls interface using Tektronics storage scopes, and continuing with the SPS controls at CERN, the accelerator community led the man-machine interface field. He noted that since the introduction of the Macintosh desktop metaphor in 1984 -- and now with Windows 3.0 for PCs -- the entire field has been subsumed by WIMPs (windows, icons, menus, and pointers). As a result, we are now the users of man-machine interfaces, not the designers.

IV. CONCLUSIONS

Two years after the Vancouver conference, similar topics are still of concern in the international controls community. More experience has been gained in the use of new interface techniques such as X-Windows and Motif, but much remains to be done. It should be very interesting to review MMI progress again in two years in Berlin.

The dichotomy between what is presented in a man-machine interface and how it is presented continues to be evident. The importance of building on higher level (e.g., accelerator) metaphors was mentioned during the discussion. While we recognize the importance of techniques for constructing man-machine interfaces, we hope that future sessions will place emphasis on the content of the interface as well.

V. REFERENCES

[1] B. Laurel, ed., The Art of Human-Computer Interface Design, Apple Computer, Inc., Addison-Wesley, 1990.

[2] K. Cahill and J. Smedinghoff, "Exploiting the X-Window Environment to Expand the Number, Reach, and Usefulness of Fermilab Accelerator Control Consoles," these proceedings.

[3] P. Antonsanti, M. Arruat, J.M. Bouche, L. Cons, Y. Deloose and F. Di Maio, "Workstations as Consoles in the CERN/PS Complex: Setting-up the Environment," these proceedings.

[4] M. Mignacco, F. Potepan, G. Surace, "The ELETTRA Man-Machine Interface," these proceedings.

[5] S. Dasgupta, D. Sarkar and C. Mallik, "Iconic Representation of Particle Beam Using Personal Computers," these proceedings.

Summary of Panel Discussion on Standards and World-Wide Sharing of Software

Organizer: P. Lucas (Fermilab)

Participants: C. Briegel (Fermilab, P.Clout (Vista), D. Gurd (SSCL), N. Kanaya (KEK), U. Raich (CERN)

It has been a dream in the accelerator community for some time that software developed for one control system be easily transferrable to and usable at another. Until recently this goal was seldom realized in practice. This has been primarily because the various control systems have been developed in-house with little standardization among them. The world of accelerators was dominated until a few years ago by very large machines constructed for doing high energy physics. The large laboratories could likewise afford large controls groups, which were able to build these complete systems from the ground up. However the accelerator scene has now shifted, with a large fraction of the new work being done at much smaller installations, installations which cannot afford the large staffs previously employed in control system production. Different approaches to this problem were outlined by P. Clout: having one or more smaller laboratories follow in the footsteps of a larger one, use of industry standards to such an extent that a significant amount of software is transferrable, or purchase of control systems from commercial vendors.

This discussion centered on the second of the points mentioned - that use of standards could foster transferability. Standards are becoming very important in the world of distributed computing as they allow the equipment of various vendors to interoperate on an integrated network. Since most accelerator control systems utilize such networks, they are indeed in the process of adopting various various standards. Among those mentioned, which have achieved to a lesser or a greater extent penetration of the accelerator field, are the Unix operating system, the X11 windowing system, the Motif presentation layer, and the TCP/IP networking protocol residing primarily on Ethernet but also on Token Ring physical layers. Most controls hardware being constructed resides in VME and to a lesser extent Multibus II, with a large installed base of Camac and its attendant driver software. The Oracle product is becoming a *de facto* standard for off-line database work, but no clear one is emerging for real-time databases. Similarly there is no standard for a microcomputer operating system, but at least there are a few commercial products being utilized, as opposed to the do-it-yourself philosophy espoused in the past.

The consensus of the discussion was that at the level of workstation console applications the prospects for shared software were good. Use of X11 and Motif have already allowed portability of some graphics widgets, an activity which is expected to continue. At levels of control lower than that of operator interaction, the prognosis is not so good. Much of the low level software of any system revolves around the database, an area in which there is no standardization as to product used or even as to the nature of the data stored.

At a less involved and more practical level it was suggested that a computer bulletin board could be initiated on which control system problems and insights could be made available to the community.

It was also noted that for software to be shared effectively it must be documented well. Although there were differences of opinion on who could best document any software and on what sort of documentation was most appropriate, there were none on the blanket statement that this is an area in which we should all strive to do better.

3rd Int. Conf. Accel. Large Exp. Phys. Control Syst. ICALEPCS1991, Tsukuba, Japan JACoW Publishing
ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S20PD03

Panel Discussion on Management of Control Systems

Don Barton (BNL), Axel Daneels (CERN), Winfried Busse (HMI), Lindsay Coffman (SSCL & DOE),
Shin-ichi Kurokawa (KEK), Rudolf Pose (JINR)

Reported by Axel Daneels (CERN)

I. INTRODUCTION

In scientific organizations one often encounters the opinion that management is a trivial activity and that project managers enjoy the easy side of the project life, far away from where the real work is. However, examples abound of projects failing to meet their objectives, running behind schedule, overrunning costs, etc., because of poor management. To several aspects which are crucial for the successful completion of a project the attention they deserve has to be paid if the project is to meet its objectives within the constraints that are imposed upon it. Whereas the engineers do things, the manager gets things done; managers are particularly concerned with:

- what is planned to be done: i.e. the product which should be delivered, in our case the control system,
- how long will the project take: i.e. schedule,
- how one will know when the project is finished: completion criteria,
- how much will it cost to implement and to maintain: i.e. the cost.

These issues can e.g. be classified in three categories respectively relating to:

- the project:
  analyse the requirements, define the quality that needs to be achieved, estimate the schedule, evaluate the cost, analyse the trade offs in order to decide e.g. whether it is preferable to phase out or to upgrade an existing system, whether one should build in house or buy commercial products, etc.,

- the logistics (hardware and software):
  what level of support should one expect during the life cycle of the project; how reliable should the system be; how maintainable; what level of safety should one reasonably expect; how much training will be requested so that the user can take up operating the new system himself, etc.,

- the technology:
  what standards should be used; are these standards likely to stay actual during the life cycle of the system; what techniques should be applied for the implementation, (e.g. computer aided software engineering - CASE -, other tools, use of advanced techniques,....); what products are available on the market that meet the requirements, etc.

II. INTRODUCING THE PANELISTS

The panelists who were invited to animate this session were selected not only on the basis of their experience in conducting control systems but also because they represent a variety of backgrounds and environments. Indeed, they all represent laboratories which are of different sizes and which operate in diverse economical and political conditions.

Each panelist introduces himself and describes briefly his current activities, the project he is concerned with, the size of the group involved in these activities and any possible "cultural" particularities of his environment that influence his activities.

- Don Barton (BNL/RHIC-AGS) heads a group of 11 program analysts, 8 electrical engineers and 9 technical support people. The group is in charge of the controls of both the Alternating Gradient Synchrotron (AGS) and the future Relativistic Heavy Ion Collider (RHIC). In addition to providing maintenance support for the running physics program at the AGS (for both protons and heavy ions) and for the commissioning of the new booster, the group soon will need to initiate the study of the RHIC controls. Besides the evolving technology, the biggest challenge stems from the sheer size of the entire accelerator complex, and consequently also of the control system. At present there are about 60 workstations and 80 to 90 multibus I crates with real time systems in a very distributed environment. The total effort invested so far is around 60 to 70 manyears.

- Winfried Busse (HMI/VICKSI) is responsible for the control system of the VICKSI facility. VICKSI is a comparatively small installation which was put into operation in 1978. Despite an upgrade program, started in 1987, the support the controls receive from the upper management of the laboratory is continuously decreasing. Originally 9 persons strong, the group is now limited to 3 people looking after the everyday running of the system and one person endorsing the entire upgrade program. It is thus no surprise that progress is very slow.

- Lindsay Coffman (SSCL & DOE) works on the DOE (Department of Energy) side of the SSC Laboratory. He is responsible for Systems Engineering across the SSC project. Coffman's office, currently 5 persons but intended to grow to 12, has to ensure that the SSC project follows the proper engineering practices: i.e. follows the modern systems engineering and management practices with discipline, adheres to current standards, follows state of the art software guidelines and development practices, delivers the proper requirement documents, etc.

- Axel Daneels (CERN/AT) involved in controls until 1990, was responsible for the development of the application software of CERN PS accelerator complex. Currently responsible for software engineering in the accelerator sector, he conducts a number of small pilot projects using CASE technology for evaluation purposes. The number of persons involved is approximately 25; they belong to different administrative entities and spend only a fraction of their time on these evaluations: this imposes some constraint on the project. In addition, the accelerator sector has little experience with new way of working which breaks quite significantly with the current habits.

- Shin-ichi Kurokawa (KEK) was working in the field of controls from 1981 to 1987. Now chairman of a division within the Accelerator Department, he is responsible for the control system whilst also coordinating the future B factory project. The controls group has 10 to 15 persons and is looking into rejuvenating the system with an eye on the new controls for the B factory. As Japanese companies are very eager to participate in physics project, such as the B factory, the laboratories have developed a skill of collaborating with industry.

- Rudolf Pose (JINR/LCTA) is Director of the Laboratory for Computing and Automation of the Joint Institute for Nuclear Research in Dubna, near Moscow. This laboratory is responsible for all computing activities, including e.g. network communication with the physics community worldwide, and controls. It has a staff of approximately 650 people. Earlier this year (1991) a new heavy ion accelerator project was decided and a small group of 12 people is currently looking into the controls of this new machine. The group will endeavour to use as much as possible existing commercial products, hardware and software.

III. ISSUES AGREED FOR DISCUSSION

Due to the limited time that is allocated to this panel, it is impossible to discuss all management issues that were mentioned in the introduction. It was therefore suggested to select a reduced number of items which were felt to be rather crucial in todays context. The participants agreed that the discussions should focus the following issues:

- Project:
  a key issue in today's project management is the dilemma of "Make vs. Buy": should we make it all in house or should we buy from industry, and if so, what should we buy ? The decision should be the result of a trade off analysis.

- Logistics:
  laboratories are increasingly concerned with their control systems rapidly becoming out of date because of the pace at which technology evolves. Increased processing power, intelligent hardware, etc., provide opportunities to explore novel, ever more sophisticated operational facilities which are difficult to achieve with the existing system. This leads to the second dilemma: "Should one upgrade the old system or embark on an entirely new project" with its corollary of having to maintenain the old system whilst implementing the new one.

- Technology:
  technology is evolving steadily: methods and techniques are coming up continuously whilst tools, e.g. computer aided software engineering (CASE) tools, are invading the market. They are intended to produce better quality systems and to assist the engineers throughout the lifecycle of their project: i.e. from the early analysis to maintenance. A study has thus to be made to select the most appropriate ones for the type of systems one is concerned with. In addition, standards are emerging and control engineers have to face the difficult choice as to which standards should be adopted.

IV. SYNTHESIS OF DISCUSSION

In the course of the debate, it appeared that the issue "make vs. buy" was raising a lot of interest: considering the time allocated to this panel, it was thus agreed to extend the discussion on this issue and to skip the one on "logistics".

- Make vs. Buy
  Similar as for other technologies (e.g. magnets, vacuum,....), it is felt that laboratories should develop a policy of market investigation for control systems also. Such a market investigation should be based on specifications resulting from a proper requirement analysis and design study. This approach would allow the laboratories to benefit from a competitive market by bidding for optimum solutions. However, one recognizes that, although it is not so difficult to write specifications and to buy off the shelf individual components, writing a definitive and complete set of requirements and proper specifications for entire systems so that they could be subcontracted, is a difficult task. It is particularly difficult to achieve in the experimental physics community because of their lack of experience, ever moving personnel, continuously changing ideas, etc. Accelerators are never finished products: as soon as the accelerator is commissioned, it becomes an R&D environment for which the controls groups have to provide the proper support. This may of course be a great challenge in itself, but is not particularly propitiuous to defining control systems in sufficient detail so that they can be bought outside. Also the time scale on which the specifications have to be carried out is in general very short and very transitory with regard to the delivery schedule. Finally, the accelerator control systems do not need to meet the same kind of severe conditions as one would expect e.g. from controls of power reactors, even if the regulatory agencies start to look at accelerators in the same way as at reactors. All these aspects are unfortunately not building up much motivation to take up the chore of producing requirements and specifications.

  The requirements should be written for a given application and not against a specific implementation: i.e. one should endeavour to subcontracting the implementation of systems or components to in-house designs, rather than buying off the shelf products which do not quite meet the requirements. A typical example of components that are difficult to find on the market are those which require specific and flexible timing treatment. On the other hand, components that are readily available on the market in general follow different standards which make it difficult for those products to work together. Significant work then has to go into integrating these components into the overall control system. Thus control system engineers have to evolve from designers to integrators.

In case complete turn key systems are bought, one if left with the problem of maintenance and upgrade: who will endorse these activities ? One should learn from the Japanese laboratories whose approach is indeed to have complete basic systems supplied by industry: i.e. the computers, the network, and the basic software, complemented by special maintenance contracts. Application programs, however, are provided in house: this is precisely an activity where tools would be of great help, not only for their implementation but also for the management of their implementation.

Industry on the other hand, may only be interested in bidding if the volume of the deal is sufficiently large. This puts the larger laboratories in a more favourable condition than the smaller ones. However, to date this is not a general policy yet even in the large laboratories like e.g. CERN, when compared with another large European Organization such as the European Space Agency (ESA) where such approach is part of their policy. This again highlights the major difference between the European and Japanese accelerator laboratories: the latter have developed a long experience of intimate collaboration with industry. Similarly, though for different reasons, laboratories in Soviet Union are bound to buy on the soviet market. Indeed, most of their budget is financed in rubles (70% of their butget) which make it difficult for those laboratories to buy from foreign countries. They thus must encourage the soviet industry in applying international standards (e.g. multibus II).

- Tools (CASE) and standards

Software engineering methods are emerging to assist the software engineer in analysing, designing and implementing large software packages. Most surprisingly, very little was said at this conference on the use of computer aided software engineering (CASE) tools. Indeed, despite their promises for better quality software and dramatic savings in maintenance, this technology is not yet widely spread in experimental physics laboratories as they tend to be in industry. Without, however, overrating the effectiveness of software houses or other industries concerned with software development in their use of CASE tools, all tend to use methods and tools as a rule. When someone joins that industry, he is instructed on the working practices and disciplines of the house. In the laboratories, however, working habits are almost opposite to everyone doing things his own way. However, one should consider that CASE methods and tools have been primarily introduced for business applications and that, as a consequence, there are not many methods and tools available for real time control applications. Among the many methodologies (structured analysis - structured design, SASD, object orientation, etc.) one needs to evaluate which are best suited for accelerator and large experimental physics control systems. This is a most difficult task as the CASE market is still unstable. New methods keep coming up at a pace that it is difficult to follow: one has not adjusted to a method that yet another one is being advocated as a panacea. These tools which complement these methods are produced by companies that are in general as new as their products; they are in general also small and have it difficult to provide adequate support.

Tools are said to be too expensive for the laboratory's budget. That argument runs into competition with the amount of manpower one has to invest in doing systems the old way. Here again it is a matter of evaluating the trade offs between the cost of buying the tools, the effort to invest in learning to use them, and the savings that can be obtained throughout the lifecycle of the control system, by using these tools. It should also be noted that in general very favourable academic discounts can be obtained for laboratories, so that even smaller laboratories could probably financially afford such tools. It is however recognized that they require a significant learning effort that is often excessively high for small groups heavily involved in every days activities. Regarding these cost considerations it is significant to notice that software houses, that are sensitive to cost and productivity, have a rather unambiguous attitude towards the use of tools: in case the money to be invested in tools competes with the cost of people, software houses tend to prefer laying off people and to buy the tools.

The significant message one should extract from preceding paragraphs is that one should at least have a method. The example of the application software for the controls of the CERN PS accelerator complex, gives an indication of the value of following a method. For that project, in the late 70-ies, structured analysis and structured design was applied extensively. At that time there were no automated tools CASE available, and the SASD was carried out with paper and pencil. However it allowed to breakdown the entire software package in "small" modules which were each completely specified so that they could be handed out to individual programmers.

Shareable software is another issue that has a strong economical impact. The appearance of object oriented (O.O.) design puts the idea of sharing software in a new light. As an example, BNL designed the controls software for the AGS booster with the use of object orientation. The software was broken down in a number of classes (in the O.O. sense) that were implemented separately. Classes could probably be used elsewhere, provided they are designed to be hardware independent.

A major step forward in the direction of sharing of software for accelerator controls could be achieved if one could agree on a standard model. This raises the problem of agreeing on standards. Control systems are costly and can not be changed frequently. They have a rather long life span and the standards which are adopted have to be in effect for several years. In 1972, it was decided to go for a distributed control system for the CERN SPS accelerator; only 20 years later, in 1992, a project is started of replacing that system. In 1972 however a distributed architecture was not so common and the choice that was made at that time might have raised a mortgage on its life cycle.

The standards that exist today (Unix, X-Windows, Motif, etc.) are primarily of use for the high level control layers where the operator interacts with the system. At the lower end, i.e. near the devices, one sees a several real time Unix operating systems emerging. They all try hard to make their way to becoming a standard. However it is still not clear how both "levels" can be tied together: e.g. what process data highway communication protocol could we standardize upon ?

Still progress in the direction of standardization is steady. At the International Conference on Accelerator and Large Experimental Physics Control Systems in Vancouver (1989) there has been a lively debate around VMS versus UNIX. Since then UNIX has won its spurs and a number of control systems are now based on UNIX. Also when analyzing the systems which were presented here, one realizes that slowly these systems tends to converge to a standard model and to the use of some common standards.

V. CONCLUSION

It is no surprise that the debates in this panel on "Management of Control System" had a strong economical flavour: the economical climate which currently prevails in most laboratories is a major concern for those involved in technologies that are not directly tied to the laboratories primary objectives, such as e.g. controls.

Laboratories should build up a habit of investigating in how far their control systems could be assembled from comprehensive packages provided by industry. This would allow them to benefit from a competitive market and to use proven products. In case entire systems are subcontracted to industry one should learn from the Japanese who also subcontract the maintenance. However, laboratories first should learn to write comprehensive specifications: a task that is particularly difficult in their ever changing experimental environment. Along these lines, the appearance of computer aided tools is an issue that is worth considering. Despite their "shakiness" and their high cost, that is often outweighed by very favourable academic discounts awarded to non commercial experimental physics laboratories, they are most valuable in producing specifications very early in the project. If it only were for that latter reason, laboratories should endeavour at entering the CASE era early and to grow together with that technology,.... or, at least, to adopt a method.

Sharing software is a dream that might become true in not so long a future. Among the new methodologies, object orientation seem to be a most promising one in view of possible sharing of software. However this issue needs a propitiuous environment of well established standards, and although standards are gradually settling, they mostly apply for the higher level control layers. The device level still has a long way to go and puts the designers in front of the difficult problem to guess which are those that are likely to be stay valid throughout the life cycle of the control system;.... we still are far from a standard model for experimental physics controls.

Finally, as technology is evolving, one recognises a shift in the activities of the control engineers. Using standards and buying "off the shelf" products, in general from different vendors, requires the control engineers to evolve from designers, to integrators who understand the art of making all these pieces work together in an coherent overall frame.

ISSUES in ACCELERATOR CONTROLS

A personal view, from a distance and in soft-focus

(Conference Summary, ICALEPCS-91)

Berend Kuiper, CERN, Geneva, Switzerland.

Dear colleagues,

The fact that I am standing here in front of you at this moment of the conference should issue two different signals. To you it should signal that the more frivolous part of the meeting has started; to me it signals that I must have reached a certain age....

The second personal comment which I want to make is that, at the lunch meeting in San Francisco, when the International Scientific Advisory Committee, ISAC, were discussing the ICALEPS-91 programme, my friend Shin-Ishi Kurokawa suggested that I should say the "closing words". Of course I was very flattered and, since that looked to me like an affair of ten minutes, I promptly accepted. By the time I received the final programme, two weeks ago, I found myself put down for a "conference summary" of 40 minutes. Of course I am still very much honoured and - noblesse oblige - I shall try ..... But what I shall really present to you, will be in the form of a, somewhat hand waving, overview of a number of main controls issues, as I think to see them, from a certain distance and in a soft-focus, with occasional side comments on what the conference gave. So it is neither of the two, or both, however you like to see it.

When attempting to make conference summaries of this kind, one is always tempted - and possibly even expected - to "discern" and then to point out the "great lines" of evolution of the subject and then to make predictions, "far sighted" if possible. Of course such an activity is jolly risky since at the beginning of any such trend, a few discernible examples and implementations of one sort, or a new product here and there, do not necessarily make a trend. By the time the developments have really taken on, however, the "great line of evolution" has become obvious to just about everyone and chances are that the trend is already approaching its end and that some other trend - at that point with hardly decodable patterns - is already infiltrating the old situation which - since it is by now known - has become comfortable and homely and - thank God - at long last more or less efficiently usable.

What, then, is there - other than the technical novelties themselves, which are so disconcertingly complex and changing - what is there to guide us in deciding our directions, to decide what course we shall choose, what we shall buy, etc. Well, ladies and gentlemen, it may sound like a platitude, but the sole constant factors in all this - on the long term - are, on the one hand, human nature with its penchant for comfort and simplicity in doing one's job and, on the other hand, the omnipresent limitations in our resources, in other words considerations of economy.

Now I am not trying to deny that some future technological breakthrough can bring an enormous benefit and change radically the way we think about our problems and that, in doing so, it may hit us from an ambush or so to speak pull the rug out under our feet. We have lived through that before. What I want to say is that this will only happen if that breakthrough brings convincingly more comfort and/or economy. Vendors are slowly learning to bring their new products and trends in a more constructive way and not any more with the sole aim of wiping out the competition and thereby possibly also the very customers they are trying to court. The reasons for this - recently more rational - behaviour is human nature and economy. The human wish for comfort blows up the software packages - systems and applications (just think of transmission protocols, graphics, etc.). The investments are then becoming so enormous that frequently learning new systems and porting software become insurmountable barriers. And creating new such barriers becomes increasingly unpopular with the clientele, who are constantly growing and, through user groups and other mechanisms, start becoming increasingly significant pressure groups. Some of the recent new technologies are therefore only taking on so rapidly since there exists a real need and because they were presented with some simultaneous adoption of standards, which warrant some degree of continuity and portability. Conversely, OS/2 (for example) has not made it since apparently there was no direct need in the field where DOS was being used (no greater comfort, just new learning) and because there were no clear signs that there will be a standard. There, where OS/2 could have made a difference, UNIX was already taking up the field and so one may say that OS/2 was too late, irrespective of its qualities which some people are praising.

But let us return to controls. In the accelerator world most controls people are still licking their wounds from the recent Wild West situation in which they were married to all kinds products whose manufacturers then either changed their mind or disappeared. We now try to start all over again and are attempting to extort money from our managements under all kind of pretexts and inventing all kind of euphemisms for the one simple message "scrap it all and build it anew". By the way, I see with pleasure that the slogan "rejuvenation" has also caught on at KEK! I assure you, dear colleagues, that it is not by mere frivolity of just wanting to have the next gadget: we are far beyond that state of mind. It is by the conviction that a new era has started in which we get really new value (meaning user comfort) and really more guarantees (meaning economy). And this is the background of many of the upgrades, revamps, rejuvenations and what have you, of which a number of interesting papers are reporting at this conference. And you may have understood, ladies and gentlemen, that with these phrases I consider having done that part of my duty which is summarising the upgrades of this conference. But this is just an introduction, as you may see.

Next come the large accelerator projects in statu nascendi like SSC, RHIC, HERA, UNK and - let us hope - LHC. Where shall they go? What will their choices be? It is interesting to speculate on the so-called trends on the basis of such more or less concrete projects. Some of them, like RHIC and LHC, are somewhat constrained by their prehistory and previous investments (economics!). They then have to choose between some continuity and homogeneity and more radical innovation. They most probably will try to steer a middle course but they will not really be able to resist the new things, because of the long project times. Then there are SSC and UNK, who both start practically without previous investments that need being safeguarded. Both are starting in a green pasture, in a certain sense, and for a certain time. SSC is confronted with a plethora of possibilities and, at least at present, apparently matching resources. UNK depends to a large extent on the USSR industrial market, which is presently still narrow, but recently they organised some hard currency influx through a collaboration with CERN. Both projects will suffer - for different reasons - from the very long project times. Sticking to one technology will prove an illusion: in the time-span of around 10 years, technology may typically change once or even twice. What, then, should guide the controls people in making their choices, which ones are meaningful and which are not, how can they avoid disaster, where shall they place their ambition, make their mark? Again the only two usable guide lines will prove to be human nature and economy of resources.

Traditionally there exists what one used to call architecture, a complex of problems on which, in the recent past, numerous experts have hotly debated and written innumerable papers at conferences and otherwise. Recently - and, dear colleagues, here comes my first controversial statement - architecture, in the conventional sense, is an issue that seems to be on its way out of the accelerator world. And we should not be too sorry for that since, to be quite honest, architecture (with all respect for my architectural friends here) is not a main issue in itself but rather it used to be a constraint which kept us from what it was really all meant for, that is controlling accelerators. It is my personal experience, which has been confirmed by looking at other labs, that the investments in architecture - which I admit were necessary in those times - have eaten up the larger fraction of the totally available controls resources and the real accelerator controls work, that which some of my friends disdainfully call "only applications", came always too little and too late. It was as if we were constantly building a piano which then kept us from really making music.

But fortunately it seems that we are on our way out of this vicious circle because of the enjoyable fact that industry, helped by the researchers in many places, are getting closer and closer to offering complete turnkey computer networks in which a very diverse collection of computers, equipment and gadgets may be simply plugged in and which then can communicate with each other in a user friendly way. The technological intestines of the computers and networks are becoming more and more invisible to the end user and even to the applications programmer. The higher end of the control system starts more and more to become a black box with a number of user related and user understandable functionalities. Although the investments in development of these products are gigantic and will remain so for some time to come, it will be the task of industries (and - conceptually - universities) and it will seize to need our (the accelerator people's) development. It will soon suffice to make a judicious choice of what building blocks to buy and then to do the integration. Moreover, the recent and continuing efforts of standardising at all levels on protocols and other interfacing conventions, makes that the plugged-in equipment and gadgets may be exchanged for newer versions, using entirely different internal technologies, which may then increase their performance but without fundamentally changing their functionality. We may actually have reached something like a "standard model" today, but the point is that the model may even evolve without wiping out our investments.

You may have noticed that at this point I think that I have dealt with architectural topology and with networking.

Of course some of you may remind me that I have not mentioned the new ideas of Rob Parker and others at SSC (and, in a sense, of Steve Magyary at LBL), that is - naively speaking - mapping the whole process address space in one huge memory of a huge and fast central computer and constantly updating those data with a fast data pump straight from the front-end, through fast multiplexed optical fiber links. Conceptually very simple, the only problem being technology. But let us not fool ourselves, this is nothing new, this is how it all began twenty years ago, this is where one starts thinking in the first place and I vividly remember my first primitive thoughts when I was parachuted into controls around 1975. At that time, that scheme was spoiled by the growing size of accelerators and the slowness of what had to be reliable transmission (CAMAC and all that). That had as a consequence, the necessity of making an explicit choice of what data really had to be fetched to the control room, since getting more than a very modest choice was simply not possible then. Today, with the new, overwhelming possibilities of data transfer, the simple "over-kill" scheme, mentioned above, may again draw within reach, even for the largest size accelerators. Then, once all the data are - physically fresh - in a central memory (preferably even the last so many of them, in a rolling buffer, well labeled with their cycle number and time-stamp), then all you need is a central machine with enough parallelism and you can do almost anything you want. And conversely, one may act back on the process globally with response times which will be vastly more "real time" than previously. But again, no illusions, the fastest global feedbacks shall always be dedicated and bypassing the central machine. In my mind there is no doubt that this scheme will now become technically feasible. The question is only whether it will be economical and - above all - whether in the large accelerator (and other) developments projects, which spread over many years and involve many separate teams, the distributed networks are not more acceptable sociologically since they allow a more natural decoupling between those teams, which in turn allows each team to proceed according to their own style and working rhythm (remember: "my car, my wife, my computer).

No doubt you will now start saying: Berend, since you are busy pooh-poohing all our cherished main issues on which generations of accelerator controls engineers have made their careers, what then do you consider a real issue? Well, one good example is certainly the man-machine interface. Not so much the workstations as such, which again may be bought from the shelf and plugged into the black box, but the way in which we are using them. It is certainly nice to have the windowing techniques on our screen and possibly on remote screens, it is nice to have many windows and to shuffle them to and fro, to the foreground or otherwise, just like papers on our desks. But the analogy goes further: many of us are used to live with a mess on our desk and having panicky moments when looking for something in the geological strata (although there are favourable exceptions, I admit). The same is of course possible on the workstation screens, only those screens have a much smaller surface area, which adds to the chaos. Nor is it attractive to constantly shift windows up and down and sideways, to the foreground or to icons, when you are in a hot operational or machine experimental phase. In the end, it is my firm belief, what we need and what cannot be replaced by windowing alone, is simply more square meters of screen and more pixels, so that a judicious choice of displays may be shown simultaneously without the need for interaction just to find the appropriate sheet. Remember that it is still infinitely simpler to flip your eyes from one screen to another than to take the mouse

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S21CS01

and call another window. So here comes my second statement: we need vastly more pixels, not developing ourselves but making the correct signs to industry, who will surely react sooner or later, since that need is not specific to accelerator controls but to human nature.

But more pixels immediately confront us with the already marginal speed of these splendid new devices. The beautiful and powerful graphics packages have vastly increased the quantity of data and code which is being manipulated and thus even our present higher end workstations could easily accept a factor of 10 improvement in throughput, without us becoming unduly spoiled. But when one aims, as I recommend, to a factor of between 10 and 100 more pixels, then it is fairly safe to say that we are still looking for a factor of around 100 in throughput before the workstations will really give us all the comfort we can use. The recommended throughput is of course not proportional to pixels, partly by algorithmic tricks, but also partly since more parallel screens displayed make for less manipulation which latter would eat most of the resources. The process data displayed will, in comparison, have more modest requirements by that time, even when we permanently display a choice of refreshing oscillograms coming up through the networks. Fine, you will say again, but all these fascinating developments can only logically be done in industry. Where can we, the accelerator controls engineers make our mark? The answer is: in the design and layout of the display and interaction surfaces, essentially a thing related to applications development, to which I shall return in a moment. It can be a fascinating work, requiring knowledge of the system, of the accelerator problem, of psychology and above all a good sense for proportions, i.e. common sense.

Now we happen to be on the subject of workstations, it occurs to me that some individuals at CERN wage an intensive action, both technically and politically, with the doctrine that PCs and DOS be the panacea for all controls matters. Let me say right away that I do not mean Alberto Pace, who is always rather balanced in his statements. And in fairness it must be said that, after Magyary's publication at Vancouver, a recent pilot project at CERN, i.e. the controls for the experiment, of which a paper is presented at this conference this morning, has handsomely and very convincingly confirmed that, if one wants, then one can, in a number of contexts, conveniently use the combination of PC - DOS - Novell - Some commercial applications - PC-I/O-cards. Earlier in this session Magyary filled us in about the present status of his system. I hope you have managed to attend: "Cela méritaitait le détour", as the Guide Michelin would say. But since the polemic persists and seems to keep both controls people and management at CERN off balance, it may be worth while dwelling a few moments on the subject. The question is whether this "PC versus Workstations" is a real issue, or even, whether using PC & DOS all-over-the-place is a breakthrough in Western thinking... Let us see...

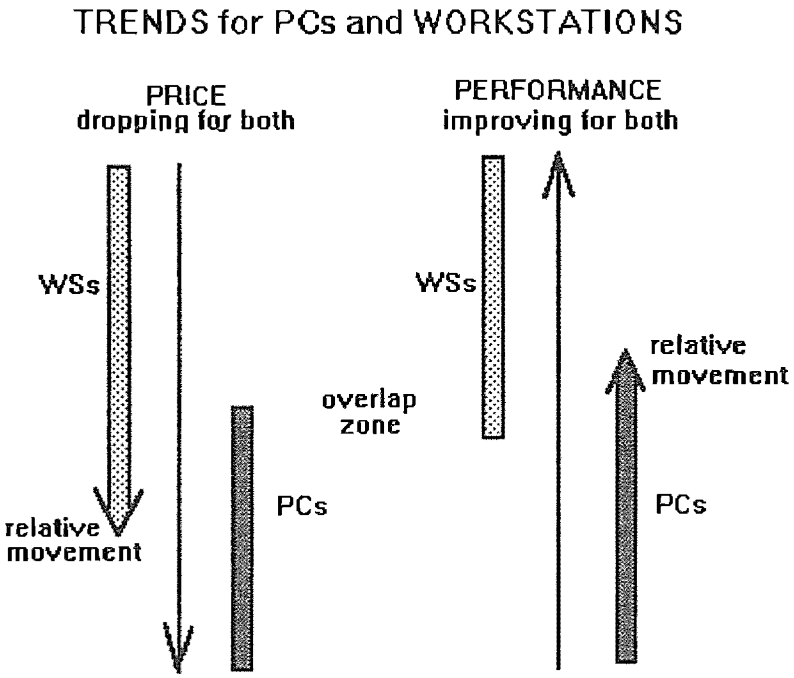
The arguments fielded in favour of using PCs all over the place, for the next generation of control systems, are more or less the following: (1) there is a plethora of high quality commercial applications software with good documentation...offered by many vendors, according to one standard...and, in some cases, with good mutual integration; (2) PCs are cheap, since they are mass produced...and there are many vendors for the same standard (conveniently forgetting the Eisa-Microchannel dichotomy!)...(3) there are excellent networking products like Novell, but also other ones...(4) there is a large collection of all sorts of plug-in cards for Input/Output and other purposes, again by different vendors. Finally, (5) using PCs for all controls allows office work and developments on accelerator controls to be done from the offices on one and the same station. All this constitutes the kit of building blocs which - remember what I said before - makes the issue of architecture redundant for accelerator controls people. One just makes the choice of what to buy, then plugs it all together, installs the commercial applications and - bingo - starts controlling.

Now that looks all fine! There are, however, also other argument to be made:
(1) The plethora of commercial software products mentioned are all running under MS-DOS, a single user operating system. For the more sophisticated networking in larger accelerator projects, however, multi tasking operating systems are a must...and in that case why not UNIX, or the like?....Moreover, the advantage of this wide choice of applications should not be overstated, because only a minute subset (the spreadsheet and one or two others) is of any practical potential use in accelerator controls... and, even so, recently the business world (who were traditionally DOS oriented) have started discovering UNIX (which previously was a hobby of universities and science freaks), so the UNIX market is now in accelerated expansion and the same software houses as referred to above are

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S21CS01

now porting their products (like Lotus 1-2-3, Wingz, Mathematica, etc..) to UNIX. Although the difference in number (all counted) is still great at the moment, the difference in available number for the accelerator controls-relevant candidates is modest and shrinking.

(2) It is true that PCs are cheap....if one takes the cheap ones. In this comparison, the PCs should be compared with the combination of workstations at the top and, say, VME based front-end processors. Taking the workstation level first, we may immediately observe that, when we compare equal things, meaning equal processing power and equal monitor screens, or, in other words, high end PCs with medium workstations, then the differences in price are not great and one should also state that today the monitor tends to be the larger part of the cost of the total package, hence by definition equal price for equal performance. Moreover the tendency is converging, as Alberto showed this morning:



At the front-end level, it is true that one PC crate is cheaper than one VME crate, but, by the time one has industrial quality and made the calculation per slot (a PC has 5 and a VME crate has 21), the difference is dwindling...and the individual Input/Output boards typically (now still unequal in price) will be converging further for similar functionality, simply by the component count and square meter price...there are no miracles at this level.

(3) No problems for networking between workstations and VME crates, the TCP/IP possibilities, with software and all, are commercially available and in future one my migrate to ISO standards and other developments, when these will become a practical reality, and all that is likely to stay entirely transparent to the user.

(4) The choice of functionalities for VME cards is already by now comparable to that for PCs (and growing fast), since from the start the industries became interested in it for their automation projects and the PC has only invaded that field coming from administration.

Finally, (5) it is now perfectly possible to call an X-window under MS-Windows on a PC screen in an office and to work on it.

So, in order to cut a long story short: the signs are that the two worlds, the PCs, on the one hand, and the combination of Workstations with VME, on the other hand, are converging and that the "PC versus Workstation" issue is a non-issue....it has been overtaken by the evolution of the market. There is of course nothing against the PC in the appropriate context (with DOS or with UNIX), but there exist no plausible arguments to use only PCs.

Now, after this frivolous interlude, let me gently move to where I think the real issues will be in the near future. It is my feeling that it will be in two broad areas: (1) one is the so-called front-end, the embedded equipment controllers (far front-end) and their connection to the upper layers (FECs) and the face these present to the applications and to the operator. (2) The second area is the applications software, in a broader sense.

Let us start looking at the front-end first. In our sort of organisations, it is more and more the tendency that the embedded intelligence is not the responsibility of the controls group and - if anything - this tendency becomes stronger. There is also some logic in that situation since there is a stronger binding between that intelligence and the device (say a power supply or beam transformer) than there is, or should be, between the control system and that local intelligence. Moreover, interfering there by the controls group would also be inefficient by the mere numbers in question (at least in the larger installations), and by the tendency of that local intelligence to grow into more or less autonomous subsystems which - in addition to their operations-oriented functionality - have an intricate internal life which is jealously guarded, and may just about be managed, by a good number of dedicated groups, each with their own electronics wizards. Thus we must henceforth resign ourselves to the idea and the fact that internal life escapes our (the controls people's) detailed understanding and control.

But we are well advised to take appropriate measures to safeguard the upper part of the control system from importing the intricacies and diversities of the far front-end, because these imports would constantly require adaptations and patches over patches, which in the end will make the system non-understandable hence unreliable. Even when not explicitly imported, these intricacies tend, over the years, to diffuse upwards, mostly under the pressure of an urgently needed functionality which would otherwise be much more complicated to implement. How then can we safeguard ourselves against these imports? Well, the old methods of agreed boundary conditions (today called "protocols") between the two domains are still as valid and as effective as ever, provided that both parties honestly use the agreed boundary conditions only and do not "hack" into the other domain. When sticking to such a discipline, both parties may arrange their internal lives more or less as they want, at least if this defensive aspect were the only one.

With somewhat more ambition, one may speculate about interchangeability of devices of the same category, without the need of changing the software above the boundary. That requires, in addition, the standardisation of the functionality of devices inside the categories.

And the ultimate speculation may be of connecting every process device through a standardised plug at any point into the control system which latter then interrogates the device and, if it is recognised (meaning that it exists in the control system's data base), configures itself accordingly and initiates the device. This needs standardisation at the electrical and connector level as well (and a few other things, of course).

Now the first level, the level of defense, has been reached long ago in hardware and there, where only hardware was concerned, the result was always quite good. But since devices today are a combination of hardware and software, the possibility, hence temptation, of cheating is much bigger and therefore the danger of the upward diffusion, mentioned above, is a real one. It can only be contained by human discipline, unless one squeezes the physical connectivity down to an unpractical level.

When pursuing the second ambition, the one of hardware interchangeability and at least partial applications portability, one must join the present endeavours around the so-called controls protocol, which was the object of the workshop this morning. This work uses the fact that an accelerator consists of only a small number of categories of process devices and that, within each category, there is a strong similitude of those functionalities which are relevant for operations (as opposed to pure engineering and service functionalities, which depend on implementation, i.e. technology and the taste of the device engineer). With some goodwill on behalf of the device engineers, these operations-relevant functionalities may be made to conform to one model. This means that the behavioural model of the device can be standardised and consequently also the software interface towards the applications programs and the appearance to the operators on the consoles. This in turn allows conserving the applications software when devices are interchanged for versions implemented in different technology. It would also allow moving devices between accelerators and also porting parts of applications from one accelerator to the other, but that is a bit more complicated and I shall come to that a bit later.

This protocols work has long suffered from conceptual and semantic misunderstandings between the parties and of laboratory-specific political difficulties, but it is now starting to make some progress inside CERN since it has finally received managerial support. But the issue is presently far from exhausted and substantial work must still be invested, including conceptual work, and it must be supported by examples of implementation, before this chapter will have reached maturity and thus the economy will start paying off the initial investments.

At the far front-end, there has since years been a hesitating, but now probably accelerating, penetration of industrial Programmed Logic Controllers, so-called PLCs, in particular for the more industrial like support services, like power, cooling water, gases, radiation protection and access control, who previously were not always integrated with the accelerator controls. It is not excluded (and there are examples) that these techniques will also diffuse into things like power supplies and vacuum, but less likely into beam instrumentation. These PLC devices come in a kit which allows configuring a range of controls functions and allows simplified applications development with a minimum of conventional coding. The material is robust and reliable and often comes with the fitting patch panel material which allows organised interfacing to the sensors and actuators in the field. Although dubbed "expensive" for many years by the accelerator controls community, there is the dawning realisation in the laboratories that, all things counted, they may in the end be cheaper than the home built controls at that level. Now it should be clear that the influx of all this new technology does not change human nature, and thus the specialised groups, will - even with such a kit - keep building their own subsystems with all the required local diagnostics and with some measure of stand alone capacity. So what I said a bit earlier about the need of well defined boundary conditions is likely to remain for some time to come, since that is largely a consequence of human nature.

Coming now to applications, it is not too surprising that, since in the past so much energy went into creating the architecture, that the applications software was always too little and too late. The causes of this were, first, the necessity to create the architecture before applications could start, secondly, that the management was not wanting to see the real cost of controls, and, last but not least, the relative programmer-unfriendliness of the system, i.e. unfriendly already for the professionals and consequently so much more so for the uninitiated operators, equipment engineers and accelerator physicists.

Although there is considerable progress since then in the basic environment and tools for program development, meaning operating systems, compilers, debuggers etc., the problem of applications development is still as severe as before since in the meantime also the user expectations have grown in step with the sophistication of the market and so one may confidently say that for any of the large accelerator controls projects, mentioned earlier, an effort of the order of 100 man*years is required for an applications package which is supposed to give some real comfort. This canonical value seems to be invariant in any transformation. Over the life cycle of the system this is considerably more. Not yet included in this are the various accompanying upgrades and rejuvenations which, depending on their ambitions, may add large fractions of the mentioned effort to the bill. At this point I am therefore already speaking about the order of 500 - 1000 man*years due in this decade, for the world's five largest accelerator centres alone. If now we also include the world's medium and small accelerators, then we may, again conservatively, triple the bill to say 2000 - 3000 man*years, so 200 - 300 Million SF or, if software houses are involved, 0.6 to 1 Billion SF. Now, try to mix into the argument the still growing level of user expectations, which is nurtured by the beautiful applications (in non-accelerator topics !), available for little money on the PC-DOS vehicle, and then there seems to be no end to that game, say 1 to 2 BSF or, to quote a nice round figure, 1 BUS$ for the decade.

Having arrived at this point one may ask the question: is there any analogy with the topic of architecture as discussed previously, is there any hope that in the next five years or so the industrial software products move equally massively into the field of applications creation, I mean applications which are relevant for the more sophisticated accelerator control?

Obviously the answer must be at least partially yes..but let us see.. There are three categories of applications software which are relevant:... First, there is a number of generic applications software packages, developed for conventional process control (say chemical plants), which are already successfully being used for controlling utilities, general services and even vacuum. Second, there exist a number of generic applications packages, which actually grew out of the accelerator field, and therefore may be called more or less accelerator oriented. Examples include: the Vsystem, the EPICS system, equally born at Los Alamos, and the CEBAF system. And then, at this conference Le Goff presented his interesting ideas about a generic control system for large physics detectors, of which I have the sneaking suspicion that it may well be adaptable and extensible to the accelerator field proper... Finally, there is Rol Johnson at Maxwell Brobeck, proposing to sell the physical applications of Fermilab (in a manner he will no doubt divulge in due course). Third, there are a few commercial programs which have not been made for that purpose but which may be used for accelerator control, in certain contexts. Examples are spreadsheets like Lotus 1-2-3, Mathematica and others, available under DOS and under UNIX.

So far so good, the industrial packages can do a good part of the job and, with some goodwill, their range of reasonable usabilty in accelerators may be somewhat extended. The programs originating from the accelerator laboratories obviously go a longer way towards our needs, but they seem not yet to be covering the field for the larger installations and more sophisticated applications. Programs of the third category often do not have the handles by which they may be easily connected to the control system, although, with some effort, it may be possible and often worth while. Programs of all three categories each cover a certain field and the ranges are to a certain degree overlapping. But data exchanges between them are in the best case awkward, which means that they lack integration. Once I have pronounced that word, let it be said loud and clearly that what we really need in the end is a degree of integration like in the MacIntosh. And then the question, which I asked previously, may be reformulated as follows: is there any hope that industry will, within the next few years, provide us with the packages which are accelerator oriented and highly integrated, like they are in the MacIntosh?

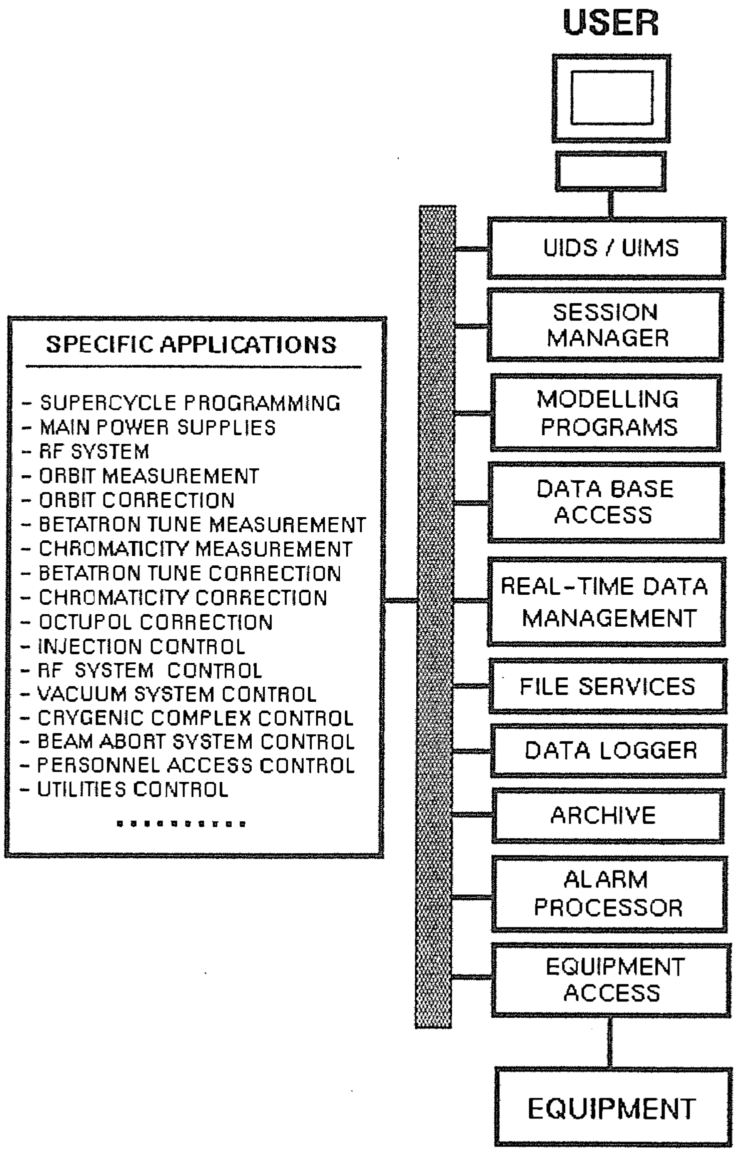
And to that question my answer would be: not jolly likely! And I think that there are several reasons. Firstly, the market is too narrow for them to deploy the huge efforts which have, for example, been the pre-requisites for the really nice and powerful software under PC-DOS. By huge efforts I mean the sum of several competing firms. Remember that the high quality was rarely achieved by the first manufacturer. For example, of the leading spreadsheets MS-Excel is the third and Borland's Quattro the fourth. And then there is of course no chance at all that industry develop our specific applications for each individual operation (unless at cost plus basis). At best they would offer us tools and a few selected generic packages, on condition that these stay close enough to the needs of the much larger industrial market. The second reason is that, in order to achieve a good coverage and relevance and quality, a large enough number of practicing accelerator people must be involved in the relevant industrial firms.... and that is not the case.

We can then do either of two things: either we resign ourselves to the situation as it is, which means that we keep muddling on with a mixture of what the market delivers today and knit it all together the best we can, accepting the limitations and frustrations, or we take the bull by the horns and try to help in the good direction. But how could we do so?

Well, first an foremost we must specify what we need. Now some people tell me that every new accelerator is different, so we cannot specify controls for a future accelerator. But we have seen that a lot of functionalities, mostly those which support the specific applications, do recur in every new accelerator in some flavour or other. There are now plenty instances to prove that point. By now that collection is called the applications environment and there are people (including me) who suspect that these program codes may be made generic and thus accelerator independent, but configurable for a range of individual accelerators. Examples of the software functionalities in the applications environment are cited in the right-hand column of the picture on the opposite page. It is from Nikolai Trofimov's talk.

The UNK Control System

Specific Applications and Application Environment



USER

SPECIFIC APPLICATIONS

- SUPERCYCLE PROGRAMMING
- MAIN POWER SUPPLIES
- RF SYSTEM
- ORBIT MEASUREMENT
- ORBIT CORRECTION
- BETATRON TUNE MEASUREMENT
- CHROMATICITY MEASUREMENT
- BETATRON TUNE CORRECTION
- CHROMATICITY CORRECTION
- OCTUPOL CORRECTION
- INJECTION CONTROL
- RF SYSTEM CONTROL
- VACUUM SYSTEM CONTROL
- CRYGENIC COMPLEX CONTROL
- BEAM ABORT SYSTEM CONTROL
- PERSONNEL ACCESS CONTROL
- UTILITIES CONTROL

....

UIDS / UIMS

SESSION MANAGER

MODELLING PROGRAMS

DATA BASE ACCESS

REAL-TIME DATA MANAGEMENT

FILE SERVICES

DATA LOGGER

ARCHIVE

ALARM PROCESSOR

EQUIPMENT ACCESS

EQUIPMENT

Note that the practical implementation of such a scheme will meet fewer technical barriers today than a few years ago, through the present trend for convergence of the architectures (Standard Model). But it is essential that this specification be done by the leading accelerator laboratories together, for several reasons. The first reason is coverage and relevance (maximum input of different requirements and experience). The second is political acceptance (those who participate tend to accept). The third is economy: only when a sufficient number of laboratories agree on these specifications, will the size of the market be large enough and will one be able to afford the price; conversely, only when the same product can be sold to a number of the bigger clients, may it be attractive enough for industry to produce it. Obviously we will never reach the price levels of Borland's Turbo Pascal at US$ 99,95 per licence, but after a few years our software bill will be vastly cheaper than it is today. And we will gain other advantages, like time to concentrate on real accelerator control problems, in contrast to spending our time on coding the same functionalities over and over again in different, incompatible flavours.

Such a collaboration could have a number of positive spin-offs. First, we are obliged to compare notes in some depth and not only by presenting each other our brilliant solutions during conferences. Second and related, the quality of the product will in general be better (if we can avoid the camel which, remember...is a horse designed by a committee). If, on the basis of these common specifications, the bigger laboratories together issue a call for tenders and negotiate a common contract, then, third, we may profit from a joint software maintenance organisation and, fourth, we may at long last get good updated documentation. Fifth, many smaller labs will then certainly join, which will stabilise the setup and improve the service. And so on...Alternatively the labs may set up a consortium of their own, but the industrial version seems to be the wiser one.

Even if we do not in the end make a common contract, the common specification exercise will be a highly interesting and rewarding one. All parties will be coming out enriched. For one thing it will become clearer what are the essentials and what is "couleur locale". I would therefore propose to make an exploratory workshop, somewhen not too late in the next year (1992!), involving around 20 or so people with experience in operation, machine physics and controls, with the aim of exploring what common ground there be....

I shall stop my speculations at this point, but you now see why it is my feeling that there will always be an interesting job in the applications (even if we keep muddling on like we do today), most certainly in the specific ones and then there will be a very challenging one in specifying, developing, updating, adapting and extending the mentioned generic software concerning the applications software environment..

Ladies and gentlemen, coming to the end of my long palaver, I wonder whether, by all those words, you may have lost the thread of he argument I am trying to make. Summarised in a few short phrases it goes as follows. A number of issues, hitherto main fields of our endeavours, which have been at the center of our attention and consumed most of our resources, will henceforth be industry driven and not any more our - the accelerator controls people's - field any more. That field, a bit broadly and hand-wavingly, may be called architecture. If you want to stay in that field, then go to industry: a lot of the advanced work will be done there. The two fields where - in my opinion -. the accelerator people will now have to concentrate their attention, where they can still be creative, where they still can make their mark, - and where they are really indispensable ! - are: first, at the front-end, in particular its connection to the henceforth industry-supplied upper architecture, and, second, - of course - the wide field of applications. But, in order to deal successfully with either or both these fields, the controls people must become much more interested in the accelerator proper (or telescope, or tokamak, ...) and that is not a bad thing. Not bad, since in this way we have moved a bit closer to what we are being paid for in the first place: that is controlling accelerators.

Thank you for your patience!

List of Authors

Bold papercodes indicate primary authors; ~~strike through/black~~ papercodes indicate **no submission received**

Institutes List

- Royer, J.-P.
- Santiago, S.
- Schinzel, J.
- Schulte, E.
- Segura-Chinchilla, N.
- Serre, Ch.
- Shering, G.
- Sicard, C.H.
- Stampfli, R.
- Steinbach, C.P.
- Strubin, P.M.
- Swoboda, D.K.
- Trofimov, N.
- Tyrrell, M.W.
- Vande Vyvre, P.
- Vascotto, A.
- Verhagen, H.
- Vos, L.

CNRS/IN2P3
Strasbourg Cedex, France

- Baumann, R.
- Kapps, E.
- Knaebel, R.
- Lutz, J.R.
- Marsaudon, J.C.
- Persigny, J.

Cockcroft Institute
Warrington, Cheshire, United Kingdom

- Owen, E.C.G.

Consorzio RFX, Associazione Euratom-ENEA sulla Fusione
Padova, Italy

- Flor, G.
- Manduchi, G.
- Piacentini, I.
- Schmidt, V.

Cornell University (CLASSE), Cornell Laboratory for Accelerator-Based Sciences and Education
Ithaca, New York, USA

- Peck, S.B.
- Rice, D.H.
- Strohman, C.R.

Czech Technical University
Prague 6, Czech Republic

- Navratil, J.

DEC-Japan
Tokyo, Japan

- Narusaka, H.

DESY
Hamburg, Germany

- Clausen, M.R.
- Rehlich, K.

Elettra-Sincrotrone Trieste S.C.p.A.
Basovizza, Italy

- Bulfone, D.
- Michelini, P.
- Mignacco, M.
- Pleško, M.
- Potepan, F.
- Surace, G.

ESO
Garching bei Muenchen, Germany

- Raffi, G.

ESRF
Grenoble, France

- Götz, A.
- Klotz, W.D.
- Meyer, J.M.

Fermilab
Batavia, Illinois, USA

- Briegel, C.I.
- Cahill, K.
- Chase, B.E.
- Franck, A.R.
- Goodwin, R.W.
- Hendricks, B.
- Holt, J.A.
- Joshel, R.H.
- Kucera, M.J.
- Lackey, S.L.
- Lucas, P.W.
- Seino, K.
- Shea, M.F.
- Shtirbu, S.
- Smedinghoff, J.G.
- Smolucha, J.
- Zagel, J.R.

GANIL
Caen, France

- David, L.
- Luong, T.T.
- Lécorché, E.
- Ulrich, M.

GSI
Darmstadt, Germany

- Krause, U.
- Riedel, C.
- Rösch, W.
- Schaa, V.RW.
- Steiner, R.

Hitachi, Ltd.
Ibaraki-ken, Japan

- Asami, K.
- Azumaishi, R.

HMI
Berlin, Germany

- Busse, W.
- Schmidt, V.

I.V. Kurchatov Institute of Atomic Energy
Troitsk, Russia

- Bulan, V.V.
- Chikovsky, V.M.
- Dudorova, E.I.

- Kartashev, A.V.
- Lusin, Yu.N.
- Nebogin, A.I.
- Oleinik, G.M.
- Papazyan, Yu.V.
- Savochkin, V.H.
- Smirnov, V.P.
- Zaitsev, V.I.

IHEP
Moscow Region, Russia
- Agababian, A.
- Agueev, A.I.
- Alferov, V.
- Brook, V.L.
- Cheng, H.J.
- Chuang, C.Z.
- Dunaitsev, A.F.
- Ekimov, A.
- Ermolin, Yu.
- Goloborodko, S.G.
- Gridassov, V.I.
- Guertsev, K.F.
- Gussak, A.A.
- Huang, H.
- Izgarshev, S.V.
- Kazarian, A.
- Khoetsian, H.
- Komarov, V.V.
- Krendelev, V.
- Lin, Q.
- Liu, D.K.
- Liu, S.-Y.
- Lu, P.
- Lukyantsev, A.
- Matevosian, A.
- Matveev, M.
- Mikheev, M.S.
- Ovcharov, S.
- Petrov, V.
- Proshin, V.M.
- Sakharov, V.P.
- Scherbakov, E.D.
- Solovyov, V.E.
- Su, Y.H.
- Sytin, A.
- Trofimov, N.N.
- Ustinov, E.A.
- Vaguine, A.I.
- Vaniev, V.
- Voevodin, V.P.
- Wang, L.Z.
- Yang, L.P.
- Yei, K.R.
- Yourpalov, V.D.
- Yue, W.
- Zelepoukin, S.A.
- Zhang, C.Z.

IMP
Lanzhou, People's Republic of China
- Chu, Z.S.
- Huang, T.H.
- Jiao, T.S.
- Li, T.Y.
- Ma, S.
- Shen, Z.
- Wang, Z.
- Zhou, X.

INFN/LASA
Segrate (MI), Italy
- Giove, D.

INFN/LNL
Legnaro (PD), Italy
- Bassato, G.
- Battistella, A.
- Bellato, M.A.
- Canella, S.

INFN/LNS
Catania, Italy
- Cuttone, G.
- Rovelli, A.

INS
Tokyo, Japan
- Katayama, T.
- Watanabe, S.
- Yoshizawa, J.

ISS
Rome, Italy
- Caccia, B.
- Dante, V.
- Lomoro, R.
- Spiriti, E.
- Valentini, S.

IUCF
Bloomington, Indiana, USA
- DuPlantis, D.
- Reid, D.
- Yoder, N.

JAEA
Ibaraki-ken, Japan
- Sugimoto, M.

JAEA/NAKA
Ibaraki-ken, Japan
- Akasaka, H.
- Aoyagi, T.
- Kawamata, Y.
- Kimura, T.
- Kurihara, K.
- Matsuzaki, Y.
- Nobusaka, H.
- Shimono, M.
- Takahashi, M.
- Terakado, T.
- Totsuka, T.
- Yagyu, J.
- Yamagishi, K.
- Yonekawa, I.

JAERI
Tokai-mura, Japan
- Hanashima, S.
- Horie, K.
- Itoh, Y.I.
- Kumahara, T.
- Shoji, T.
- Tsukihashi, Y.
- Yoshikawa, H.

JAERI-RIKEN/Spring-8 Project Team
Tokyo, Japan
- Kumahara, T.
- Masuda, T.
- Wada, T.
- Yonehara, H.
- Yoshikawa, H.
- Zhen, W.

JINR
Dubna, Moscow Region, Russia
- Kotov, V.M.
- Pose, R.A.

KEK
Ibaraki, Japan
- Abe, I.
- Adachi, T.
- Ajima, Y.
- Akiyama, A.
- Arakawa, D.A.
- Araki, S.
- Furukawa, K.
- Hiramatsu, S.
- Holt, J.A.
- Ieiri, M.
- Ishida, T.
- Ishihara, N.
- Ishii, H.
- Ishii, K.
- Kadokura, E.
- Kakihara, K.
- Kamikubota, N.
- Kanaya, N.
- Kato, Y.
- Katoh, T.
- Kawakubo, T.
- Kawamoto, T.
- Kishiro, J.
- Komada, K.
- Kudoh, K.
- Kurokawa, S.-I.
- Matsumoto, S.
- Mikawa, K.
- Mimashi, T.
- Minakawa, M.
- Naitoh, T.
- Nakagawa, H.
- Nakahara, K.
- Nigorikawa, K.N.
- Noumi, H.
- Ogawa, Y.
- Ohsawa, S.
- Otake, Y.
- Pak, C.O.
- Sato, H.
- Satoh, Y.
- Shidara, T.
- Someya, H.
- Sueno, T.
- Suzuki, Y.
- Takasaki, M.
- Takeda, S.
- Tanaka, K.H.
- Toda, M.
- Toyama, T.
- Urakawa, J.
- Yamanoi, Y.
- Yokota, M.

KFKI
Budapest, Hungary
- Vesztergombi, G.

KHI
Kobe, Japan
- Ishihara, K.
- Kubota, T.
- Nakabayashi, S.
- Yasuda, K.

LANL
Los Alamos, New Mexico, USA
- Björklund, E.
- Burns, M.J.
- Callaway, T.
- Carr, G.P.
- Cohen, S.
- Dalesio, L.R.
- Harrington, M.
- Hill, J.O.
- Kerstiens, D.M.
- Kozubal, A.J.
- Kubicek, D.
- Poore, R.
- Scarborough, W.K.
- Schaller, S.
- Schultz, D.
- Stuewe, R.

LBNL
Berkeley, California, USA
- Chin, M.J.
- Cork, C.W.
- Fahmie, M.P.
- Lancaster, H.
- Magyary, S.
- Molinari, P.
- Nishimura, H.
- Robb, A.
- Timossi, CA.
- Young, J.

LNF-INFN
Frascati, Italy

- Di Pirro, G.
- Milardi, C.
- Serio, M.
- Stecchi, A.
- Trasatti, L.

Mitsubishi Electric Corporation
Tokyo, Japan
- Ishida, T.

MRI
Moscow, Russia
- Filipchicov, L.L.
- Frolov, I.
- Ivanov, Yu.S.
- Konovalov, V.A.
- Murin, B.P.
- Silin, A.
- Vaguine, A.I.

MRTI RAS
Moscow, Russia
- Kabanov, V.S.

MSL
Stockholm, Sweden
- Engström, M.
- Starker, J.

MSU
Moscow, Russia
- Chepurnov, A.S.
- Gribov, I.V.
- Morozov, S.Yu.
- Shumakov, A.V.
- Zinoviev, S.V.

NAO-NRO
Minamisaku, Nagano, Japan
- Iwashita, H.
- Kanzawa, T.
- Kawashima, S.
- Morita, K.I.
- Nakajima, H.
- Nishio, M.
- Takahashi, T.
- Torii, C.

NIFS
Chikusa-ku, Nagoya, Japan
- Kaneko, H.
- Motojima, O.
- Taniguchi, Y.
- Yamazaki, K.

NIIEFA
St. Petersburg, Russia
- Mozin, L.V.
- Shelekhov, V.A.
- Vasiljev, N.D.

NIKHEF-K
Amsterdam, The Netherlands
- Heubers, W.
- Tummers, B.J.

NIRS
Chiba-shi, Japan
- Asami, K.
- Azumaishi, R.
- Endo, M.
- Hirao, Y.
- Itano, A.I.
- Kanai, T.
- Kanazawa, M.
- Kawachi, K.
- Kohno, T.
- Minohara, S.
- Morii, Y.
- Narusaka, H.
- Noda, K.
- Sato, K.
- Soga, F.
- Sudou, M.
- Takada, E.
- Tsuzuki, N.

NSRRC
Hsinchu, Taiwan
- Chen, C.J.
- Chen, J.
- Jan, G.J.
- Wang, C.S.

NTUT
Taipei, Taiwan
- Wang, C.S.

PAL
Pohang, Kyungbuk, Republic of Korea
- Chang, S.-S.
- Choi, S.J.
- Huang, J.Y.
- Huang, J.
- Kim, J.-H.
- Kim, M.S.
- Kirn, J.H.
- Lee, J.-W.
- Lee, J.W.
- Lee, J.
- Namkung, W.
- Park, M.K.
- Park, S.Y.
- Won, S.C.

Plasma Research Laboratory, University of Malaya
Kuala Lumpur, Malaysia
- Chew, A.C.
- Lee, S.
- Saw, S.H.

Power Reactor and Nuclear Fuel Development Corporation, Oarai Engineering Center
Oarai, Japan
- Endou, A.
- Himeno, Y.
- Miki, T.

RAL
Chilton, Didcot, Oxon, United Kingdom
- Barton, C.J.
- Brownless, D.M.
- Kerr, J.C.
- Mannix, R.P.

RAS/INR
Moscow, Russia
- Bragin, S.
- Esin, S.K.
- Ilinsky, N.G.
- Reinhardt-Nickulin, P.I.
- Senichev, Y.

Saint Petersburg State University
Saint Petersburg, Russia
- Edamenko, N.S.
- Ovsyannikov, D.A.
- Zhabko, A.P.

SERC
Daresbury, Warrington, Cheshire, United Kingdom
- Martlew, B.G.
- McCarthy, M.
- Rawlinson, W.R.

SHI
Tokyo, Japan
- Aoki, K.
- Ohnishi, K.

SLAC
Menlo Park, California, USA
- Allison, S.
- Clark, S.
- Grillo, A.A.
- Gromme, T.
- Hendrickson, L.
- Himel, T.M.
- Humphrey, J.
- Hutchinson, D.A.W.
- Kang, H.
- Krauter, K.E.
- Lee, M.J.
- Millsom, D.
- Nelson, D.
- Olsen, J.D.
- Phinney, N.
- Sanchez-Chopitea, L.
- Sass, R.C.
- Shoaee, H.
- Spencer, N.C.
- Underwood, K.K.
- White, G.R.
- Zelazny, S.

Space and Laser Communication Development Division, NEC Corporation
Yokohama, Japan
- Futagawa, N.
- Nishikawa, K.
- Takabayashi, T.

SSCL
Dallas, TX, USA
- Allen, M.E.
- Bourianoff, G.
- Christiansen, M.
- Coffman, L.
- Dohan, D.
- Gurd, D.P.
- Haenni, D.J.
- Hunt, S.M.
- Low, K.
- Trahern, C.G.
- Zhou, J.

Sumitomo Electric Industries Ltd., Yokohama Laboratory
Yakohama, Japan
- Tanaka, S.

TMEIC
Tokyo, Japan
- Morii, Y.

Tohoku University, School of Science
Sendai, Japan
- Kitamura, M.
- Mutoh, M.
- Oonuma, T.
- Shibasaki, Y.

Toshiba Corporation
Tokyo, Japan
- Morii, Y.
- Shigeta, M.
- Toyoda, E.
- Tsuzuki, N.
- Yagi, T.
- Yamagishi, T.
- Yamazaki, C.

Toshiba Mitsubishi Electric Industrial Systems Corporation
Tokyo, Japan
- Tsuzuki, N.

TRIUMF
Vancouver, Canada
- Dale, D.
- Ludgate, G.A.
- Wilkinson, N.A.

TSL
Uppsala, Sweden
- Gajewski, K.J.
- Johansson, O.
- Thuresson, L.

UCD
Davis, California, USA
- Rouse, F.

UCLouvain-FYNU
Louvain-la-Neuve, Belgium
- Martou, J.L.
- Nemry, M.
- Ninane, A.H.
- Somers, F.

USTC
Hefei, Anhui, People's Republic of China
- Liu, Y.
- Xu, W.

USTC/NSRL
Hefei, Anhui, People's Republic of China
- Yao, C.-Y.

VECC
Kolkata, India
- Dasgupta, S.
- Mallik, C.
- Sarkar, D.

VISTA
Los Alamos, NM, USA
- Clout, P.N.
- Geib, M.
- Westervelt, R.

YerPhI
Yerevan, Armenia
- Kazarian, A.
- Khoetsian, N.
- Toumanian, A.
- Zapolski, N.

PARTICIPANT LIST OF ICALEPCS '91 (NOV. 11 - NOV. 15, 1991)

Abe, Isamu *(KEK)*
Akiyama, Atsuyoshi *(KEK)*
Alferov, Vladimir *(IHEP-Protvino)*
Allen, Mike *(SSCL)*
Amtey, Sharad *(SSCL)*
Araki, Sakae *(KEK)*
Bailey, Roger *(CERN)*
Baribaud, Guy *(CERN)*
Barton, Carole Joan *(RAL)*
Barton, Donald S. *(BNL)*
Batskikh, G. *(MRI)*
Beetham, Christopher Gary *(CERN)*
Blumer, Thomas *(PSI)*
Bork, Rolf *(SSCL)*
Bourianoff, George I. *(SSCL)*
Briegel, Charles I. *(Fermilab)*
Bulfone, Daniele *(INFN-Trieste)*
Burns, Alan James *(CERN)*
Busse, Winfried *(HMI)*
Cahill, Kevin *(Fermilab)*
Canella, Stefania *(INFN-LNL)*
Cavallari, Giorgio *(CERN)*
Chang, Suk Sang *(POSTECH)*
Chepurnov, Alexander S. *(MSU)*
Chohan, Vinod *(CERN)*
Ciapala, Edmond *(CERN)*
Ciriani, Paolo *(CERN)*
Clausen, Matthias *(DESY)*
Clifford, Tom S. *(BNL)*
Clout, Peter Norman *(VISTA)*
Coffman, Lindsay *(SSCL)*
Crowley-Milling, Michael
Cuttone, Giacomo *(INFN-LNS)*
Dalesio, Leo Robert *(LANL)*
Daneels, Axel Jean *(CERN)*
Dasgupta, Subrata *(VECC)*
Dimaio, Franck *(CERN)*
Di Pirro, Giampiero *(INFN-LNL)*
Dickey, Carl Edward *(SSCL)*
Dohan, Don A. *(SSCL)*
Dunaitsev, Anatoly F. *(IHEP-Serpukhov)*
Emura, Katsuji *(Sumitomo E.I.-Osaka)*
Engström, Mats *(MSI)*
Ermolin, Yuri *(IHEP-Serpukhov)*
Frolov, I. *(MRI)*
Furukawa, Kazuro *(KEK)*
Gajewski, Konrad Jan *(TSL)*
Giove, Dario *(INFN-LASA)*
Götz, Andy *(ESRF)*
Grant, Peter Angus *(TRIUMF)*
Gribov, Igor V. *(MSU)*
Gurd, David P. *(SSCL)*
Hacker, Uli *(KFA)*
Haenni, David Richard *(SSCL)*
Hama, Hiroyuki *(IMS)*
Hanashima, Susumu *(JAERI-Tokai)*
Harada, Shunji *(MELCO-NFDD)*
Heinze, Wolfgang *(CERN)*
Hendricks, Brian *(Fermilab)*
Hendrickson, Linda Joyce *(SLAC)*
Heubers, Wim *(NIKHEF)*
Hien, Pham Zuy *(CANT)*
Hsu, K.T. *(SRRC)*
Huang, Jung-Yun *(POSTECH)*
Humphrey, John (Rusty) *(SLAC)*
Hunt, Steve *(SSCL)*
Iselin, Christoph F. *(CERN)*
Ishii, Kazuhiro *(KEK)*
Jan, Gwo-Jen *(SRRC)*
Jiao, Tian-shu *(IMP)*
Joshel, Robert H. *(Fermilab)*
Kabe, Seiji *(KEK)*
Kadokura, Eiichi *(KEK)*
Kamikubota, Norihiko *(KEK)*
Kanaya, Noriichi *(KEK)*
Kanazawa, Mitsutaka *(NIRS)*
Kaneko, Hiroshi *(NIFS)*
Kase, Masayuki *(RIKEN)*
Katoh, Tadahiko *(KEK)*
Katsura, Tomotaro *(KEK)*
Kawakubo, Tadamichi *(KEK)*
Kawamoto, Takashi *(KEK)*
Kijima, Yuko *(MELCO-NFDD)*
Kimura, Toyoaki *(JAERI-Naka)*
Kimura, Yoshitaka *(KEK)*
Kishiro, Jun-ichi *(KEK)*
Kitamura, Masaharu *(LNS)*
Klotz, Wolf-Dieter *(ESRF)*
Knott, Martin John *(ANL)*
Kohno, Toshiyuki *(NIRS)*
Komarov, Vladimir *(IHEP-Protvino)*
Kouno, Satoshi
Kraimer, Martin R. *(ANL)*
Kubota, Tetsuya *(Kawasaki H.I.)*
Kudo, Kikuo *(KEK)*
Kuiper, Berend *(CERN)*
Kumada, Masayuki *(NIRS)*
Kumahara, Tadashi *(JAERI-Tokai)*
Kuper, Edward *(INP-Novosibirsk)*
Kurokawa, Shin-ichi *(KEK)*
Lécorché, Eric *(GANIL)*
Lee, Martin Joe *(SLAC)*
Le Goff, Jean-Marie H. *(CERN)*
Liu, Shi-Yao *(IHEP-Beijing)*
Lucas, Peter Wayne *(Fermilab)*
Ludgate, George Arthur *(TRIUMF)*
Lukyantsev, Alexander F. *(IHEP-Serpukhov)*
Luong, Tam T. *(GANIL)*
Lutz, Jean-Robert *(IN2P3-CNRS)*
Magyary, Steven Bela *(LBL)*
Mannix, Rbert Patrick *(RAL)*
Marsaudon, Jean-Claude *(IN2P3-CNRS)*

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991

Masuda, Takemasa *(RIKEN)*
Matsutomi, Shokichi *(Toshiba-Fuchu)*
Matsuzaki, Yoshimi *(JAERI-Naka)*
McDowell, William P. *(ANL)*
Mertens, Volker *(CERN)*
Miki, Tetsushi *(PNC-Oarai)*
Mimashi, Toshihiro *(KEK)*
Minohara, Shinichi *(NIRS)*
Miura, Takeo *(Nippon Eng.)*
Morita, Koh-Ichiro *(NAO-NRO)*
Mozin, Igor *(Efremov)*
Mueller, Klaus-Dieter *(KFA)*
Murai, Katsuji *(Hitachi-Omika)*
Murin, Boris P. *(MRI)*
Mutoh, Masakatsu *(LNS)*
Nakabayashi, Shiro *(Kawasaki H.I.)*
Nakagawa, Hidetoshi *(KEK)*
Nakamura, Takeshi *(RIKEN)*
Navratil, Jiri *(CTU)*
Nguyen, Huan T. *(SSCL)*
Nigorikawa, Kazuyuki *(KEK)*
Ninane, Alain H. *(UCL)*
Nishimura, Hiroshi *(LBL)*
Nishio, Masanori *(NAO-NRO)*
Ogata, Hiroshi *(RCNP)*
Okumura, Susumu *(JAERI-Takasaki)*
Otake, Yuji *(KEK)*
Owen, Edward Charles *(Daresbury)*
Oyamada, Masayuki *(LNS)*
Pace, Alberto *(CERN)*
Pak, Cheol On *(KEK)*
Peters, Franz *(DESY)*
Pose, Rudolf *(JINR)*
Potepan, Franco *(INFN-Trieste)*
Protasov, Yuri S. *(MSU)*
Rabany, Michel *(CERN)*
Raffi, Gianni *(ESO)*
Raich, Ulrich *(CERN)*
Rausch, Raymond *(CERN)*
Rawlinson, William Robert *(Daresbury)*
Reinhard-Nickulin, Petr I. *(INR)*
Riedel, Larry Dean *(SSCL)*
Saban, Roberto *(CERN)*
Sarkar, Surajit *(SSCL)*
Saw, Sor-Heoh *(Univ. Malaya)*
Schaa, Volker R.W. *(GSI)*
Schaller, Stuart *(LANL)*
Schekochihin, Arkady V. *(INP-Protvino)*
Schimmel, Fred *(NIKHEF)*
Schmidt, Volker *(IGI)*
Serre, Christian *(CERN)*
Serre, Claudia *(IN2P3-Paris)*
Shea, Michael F. *(Fermilab)*
Shering, George Craig *(CERN)*
Shibasaki, Yoshinobu *(LNS)*
Shibata, Shinkichi *(TCT)*
Shirakawa, Akihiro *(KEK)*
Skarin, Igor A. *(INP-Protvino)*
Skelly, Joseph Francis *(BNL)*
Starker, Josef *(MSI)*
Stecchi, Alessandro *(INFN-LNF)*
Steiner, Rudolf *(GSI)*
Strohman, Charles Ralph *(Cornell)*
Stuewe, Robert B. *(LANL)*
Sugimoto, Masayoshi *(JAERI-Tokai)*
Suwa, Shigeki
Suzuki, Yoshihiro *(KEK)*
Takada, Eiichi *(NIRS)*
Takahashi, Minoru *(JAERI-Naka)*
Takahashi, Mitsuyuki *(IHI)*
Takebe, Hideki *(RIKEN)*
Takeda, Shigeru *(KEK)*
Tanabe, Eiji *(AET-Japan)*
Tanaka, Shigeru *(Sumitomo E.I.-Yokohama)*
Tanaka, Hitoshi *(RIKEN)*
Trahern, Charles Garrett *(SSCL)*
Trasattil, Luciano *(INFN-LNF)*
Trofimov, Nikoly Nik. *(IHEP-Serpukhov)*
Tsybin, Alexander S. *(MSTU)*
Tumanian, Arsen Rafael *(YPI)*
Tyrrel, Mark William *(CERN)*
Urakawa, Junji *(KEK)*
Utsumi, Masafumi *(MELCO-NFDD)*
Vaguine, Alexei *(MRI)*
Vande Vyvre, Pierre *(CERN)*
Wada, Takeshi *(RIKEN)*
Wang, Zhen *(RIKEN)*
Watanabe, Shin-ichi *(INS)*
Weng, C.S. *(SRRC)*
Wilkinson, Neil Adam *(TRIUMF)*
Won, Sangchul *(POSTECH)*
Xu, Weimin *(HESYRL)*
Xue, Jing-Xuan *(IHEP-Beijing)*
Yagi, Takao *(Toshiba-Fuchu)*
Yamada, Kohji *(NKK)*
Yamamoto, Noboru *(KEK)*
Yamazaki, Kozo *(NIFS)*
Yao, Chih-Yuan *(HESYRL)*
Yonekawa, Izuru *(JAERI-Naka)*
Yasunobu, Seiji *(Hitachi-Ozenji)*
Yoshikawa, Hiroshi *(JAERI-Tokai)*
Zagel, James R. *(Fermilab)*
Zaitsev, V.I. *(Kurchatov)*
Zlelepoukin, Sergei *(IHEP-Protvino)*