NEW FRONT-END COMPUTERS BASED ON LINUX-RTAI AND PPC

D. Bulfone, R. De Monte, V. Forchi', G. Gaio, M. Lonza, L. Pivetta
Sincrotrone Trieste, Trieste, Italy

Abstract

The ELETTRA control system front-end computers are presently based on 68k VME boards and the OS-9 operating system. In view of the construction of the new booster injector and of a smooth upgrade of the existing control system, PowerPC VME boards running Linux have been adopted. The new platform provides reliability, performance and flexibility, while the RTAI (Real Time Application Interface) extension offers, where necessary, satisfying real-time capabilities that compete well with those of the most popular real-time operating systems. This article describes the main issues associated to the choices we made and presents an example of application.

INTRODUCTION

ELETTRA, the third generation synchrotron light source based in Trieste, Italy, is in its tenth year of operation. The existing control system, designed in the early `90s using state-of-the-art technologies, features a three layer architecture with Unix workstations at the higher layer and VME bus systems with 68k CPU boards running the OS-9 operating system at the middle and lower layers [1]. While workstations and the middle layer communicate via an Ethernet network, the field-bus MIL-1553B is employed to connect the middle to the lower layer computers.

In the last few years some of the hardware components, especially CPU and fieldbus communication boards, became obsolete and even unserviceable. A smooth upgrade of the control system with a two-layer architecture started three years ago. The new front-end computers were based on 68040 CPU boards still running the OS-9 operating system but connected directly to Ethernet [2]. Since then twelve new low-level computers have been installed to control new accelerator devices and seventeen of the old front-end computers have also been upgraded with the new CPU boards allowing elimination of three out of ten of the middle layer computers.

This minor upgrade, although necessary for obsolescence of the hardware, has been the first step towards the definition of a completely new control system. The incoming construction of a booster injector for ELETTRA and the development of a fast orbit feedback system have definitely stimulated a new design, which benefits from the experience gained with the new architecture while using the latest technologies [3].

FRONT-END HARDWARE

In a two-layer architecture, the front-end computers must provide the hardware interface to the controlled equipment through I/O signals, enough processing power to run the proper control programs and a robust network interface through which any client can access in an exhaustive object oriented way the controlled device.

VME or cPCI?

Modular systems based on standard industrial buses are still a good choice that gives the flexibility to adapt to any requirement. The natural alternative to VME is CompactPCI (cPCI), that although offering a better performance and hot swap capabilities suffers from single mastering capabilities and slot number limitation. Some years ago cPCI seemed to surpass VME in the overall market, however, today VME is still strong and its future is bright thanks to military and legacy markets which sustain it because of its long life span. The cPCI market has instead suffered from the collapse of the communication market and the future of cPCI as the successor of VME now seems also to be threatened by serial bus technologies (switched fabric). These are however still too young to be adopted. Our final choice for the new control system is in favor of VME (VME64x) that still provides good performance and allows to re-use part of the legacy hardware.

CPU Board

One of the goals in designing the new front-end computers has been the adoption of a single CPU board able to embrace several applications ranging from standard "slow" controls to fast orbit feedback systems. This approach could seem rigid but has many advantages that avoid the diversification of hardware and ease the management and maintenance of both hardware and software. A market survey was carried out two years ago taking into account overall performance, quality of manufacturing and life span in terms of market availability and servicing. The board selected is the Motorola MVME5100, a single-slot single board computer based on the PowerPC processor. Presently the board features a MPC7410 (G4) microprocessor running at 400 MHz and 512 MB of RAM. A PCI local bus accommodates two on-board PMC (PCI Mezzanine Card) slots with front or rear I/O, whereas a PMC stackable carrier adapter allows up to four additional PMC slots. A PCI to VME bridge provides the VME64 functionality.

The PowerPC G4 microprocessor features outstanding performance achievement at a considerably lower clock rate with respect to x86 microprocessors resulting in less power consumption and easier board level integration, making the microprocessor more suitable for embedded applications. Moreover, with its AltiVec technology, G4 can be deployed also for Digital Signal Processing (DSP) applications, where expensive dedicated DSP boards were the only option till a few years ago.

I/O Hardware

Different types of I/O are foreseen for interfacing the control system to the equipment: analog/digital signals, serial lines (RS232, RS422, etc.), GPIB, CAN-bus and video signals. Apart from a given number of custom VME boards for special applications like beam diagnostics, timing system or I/O in extremely noisy environments, commercial standard solutions will be adopted.

Mezzanine technology is attractive for the flexibility it offers. Industry Pack (IP) and PMC are the preferred standards, which are pretty complimentary to each other. IP is a widely available low cost solution for non-intelligent I/O not requiring high performance; VME IP carriers can host up to four modules allowing to customize the I/O board by adding one or more modules. PMC is used where high performance or intelligent I/O is needed. In this case the two+four PMC slots of the MVME5100 board can be used, resulting in extremely compact systems. Both IP and PMC modules feature I/O on the P2 VME connector providing clean connections and cabling via rear side transitions modules.

SYSTEM SOFTWARE: LINUX

The opportunity of using the GNU/Linux operating system in the front-end computers is quite attractive. It is open source, free of charge, actively developed, reliable, efficient and does not come from a single software supplier. A rich set of ready to be used application software exists for GNU/Linux, as well as a complete development environment featuring all the main programming languages such as C, C++, Java, Python, etc. Several incarnations of the CORBA object oriented technology are available for Linux. In addition, developing device drivers in Linux is fairly easy and can be done in C language. Most of I/O hardware manufacturers support Linux and the device driver source code is almost always available. After a thorough analysis of the requirements of the front-end computers we concluded that Linux is suitable for this purpose and have decided to adopt it.

The present installation is based on YellowDog Linux 2.1 and kernel version 2.4.18 for the Motorola MVME5100 board developed by MontaVista [4]. An upgrade to YellowDog 3.0 is planned in the near future. The development systems as well as the field computers are diskless machines booting from a central server via the TFTP network protocol.

Linux is not real-time because it was originally designed to privilege throughput over responsiveness and determinism. Although current CPU performance provides fast response and extensive computing power, the Linux kernel cannot guarantee real-time behaviour. Two main approaches have been proposed to add real-time capabilities to Linux.

The first improves the responsiveness by modifying the Linux kernel and scheduler. Applications are developed and run in a standard Linux environment with improved deterministic behaviour, but the latency is still not guaranteed. The second approach consists of inserting a second real-time micro-kernel and running Linux as a "low priority" task. The real-time kernel takes control of the interrupt management and dispatches the requests to the real-time Interrupt Service Routines (ISR) or to standard Linux handlers. Processes and applications that do not require hard deterministic response times can be run in Linux, whereas the real-time kernel executes the tasks with demanding latency.

Real Time Application Interface (RTAI) [5], a completely free implementation of the second approach developed at Milan polytechnic, has been chosen at ELETTRA. RTAI features multi-list schedulers, extended POSIX APIs, memory management, FIFOs, semaphores, mailboxes, watchdog, etc. The MontaVista *linuxppc* kernel source tree has been patched using RTAI 24.1.8 from DENX [6]. Being distributed under the GPL and LGPL licenses, RTAI is not covered by patents.

REAL-TIME PERFORMANCE BENCHMARKS

Performance tests using the setup described below have been carried out to characterize the system response time. A VME board generates interrupts on the bus at each rising edge of a 10 kHz clock. On the CPU, a RTAI Interrupt Service Routine (ISR) first asserts a digital signal using a VME I/O board and then releases two semaphores that unlock two RTAI threads waiting on them. Both threads, which are scheduled by RTAI one after the other, generate another signal through the digital I/O board. The three digital signals as well as the 10 kHz clock signal are acquired by a logic state analyzer with statistics capabilities. The system has been heavily loaded with tasks involving a large volume of interrupts: multiple "flood pinging", several processes generating network activity and high-speed serial I/O traffic.

The following measurements have been carried out:

- Delay between the clock rising edge and the first digital signal (ISR interrupt latency).
- Delay between the clock rising edge and the second digital signal (thread interrupt latency).
- Delay between the second and third digital signals (context switching).

For each of them a histogram including roughly two millions measurements has been generated: the results are shown in Fig. 1.

APPLICATION EXAMPLES

As part of a general ongoing installation program, higher performance electron Beam Position Monitors (BPM) [7] have been installed in two ELETTRA straight sections either side of the Insertion Devices (ID). They take advantage of digital detection electronics to provide fast beam position measurements with sub-micron precision. For each BPM, a digital detector VME board supplies the X and Y position data at an 8 kHz rate via its slave VME interface using a FIFO, which can be

programmed to generate VME interrupts when full, empty, almost-full or almost-empty.

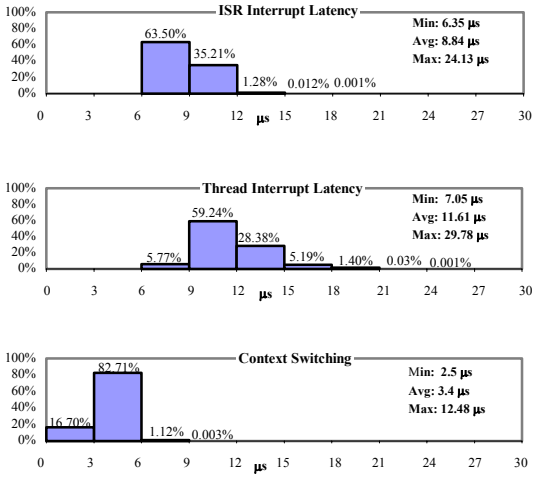

Figure 1: Statistics of the response times of a PowerPC VME board running Linux/RTAI.

Three applications developed in Linux/RTAI make use of the fast position data furnished by the digital BPMs at each ID location. The first is a fast local orbit feedback system running at 8 kHz that employs four corrector magnets to stabilize the electron beam orbit at the ID centre on the basis of the position measured by the two BPMs [8]. The second is a calibration system that provides a lookup table used by a feed-forward correction system compensating for the orbit distortion produced by an Electromagnetic Elliptical Wiggler (EEW) operated at high switching frequency [8]. The third is a general-purpose acquisition/monitoring system that can manage a number of linear/circular buffers where data from the BPMs can be recorded at different frequencies.

The current implementation allows these applications to run in parallel on the same front-end computer. One digital detector board per VME crate is programmed to generate a VME interrupt every time its FIFO is not empty, thus at 8 kHz repetition rate. In order to allow multiple "consumer" applications to use the data from the BPMs, a device driver module has been developed in RTAI (Fig. 2). When the interrupt from the VME bus is received, the driver ISR reads the data from all the digital detector FIFOs and releases a semaphore unlocking a RTAI thread that calls a list of consumer routines. Any RTAI module requiring the BPM data must register to this service by adding its acquisition routine into the list.

A dedicated server running in Linux is associated to each of the three applications. Each server exchanges data with the corresponding real-time module by means of a shared memory and provides a network interface for the application.

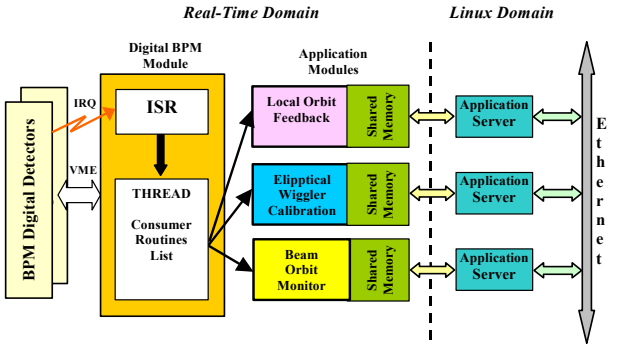

Figure 2: Block diagram of the software structure of a "multiple-consumer" application using position data from digital BPMs.

Following the philosophy already described only those parts of the applications that require deterministic latency, consisting of restricted parts of the code, run in the real-time domain. The rest can run in Linux and benefit from a powerful and reliable environment. All three real-time applications can comfortably run together on the same CPU. Measurements show that, in the worst case, they occupy less than 20% of CPU time, the rest is available to Linux.

CONCLUSION

The new front-end computers of the ELETTRA control system, based on PowerPC VME boards running the GNU/Linux operating system with the RTAI extension, combine the advantages of working in a Linux environment and enough real-time capabilities to meet the requirements of the accelerator control system.

The new front-end computers have been recently employed to develop a fast local orbit feedback system installed in two ID straight sections. The commissioning and the first operation of the feedback system have proven reliability and excellent overall performance.

REFERENCES

[1] D. Bulfone, "Status and Prospects of the ELETTRA Control System", Nuclear Instrumentation and Methods A 352, p. 63, 1994.

[2] M. Lonza et al., "New Low Level Controls for the ELETTRA Linac", ICALEPCS'99, Trieste, October 1999.

[3] C. Scafuri et al., "Toward the ELETTRA New Injector Control System", these proceedings.

[4] http://www.mvista.com/

[5] http://www.aero.polimi.it/~rtai/index.html

[6] http://www.denx.de

[7] M. Ferianis et al., "The Low Gap BPM System at ELETTRA: Commissioning Results", DIPAC'01, Grenoble, May 2001.

[8] D. Bulfone et al., "Exploiting Low-Gap Beam Position Monitors in Orbit Stabilization Feedback and Feed-Forward Systems at ELETTRA", Journal of the Japanese Society for Synchrotron Radiation Research, Vol.16 No.4, July 2003.