ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUPHA210

A BUNCH-SYNCHRONIZED DATA ACQUISITION SYSTEM FOR THE EUROPEAN XFEL ACCELERATOR

T. Wilksen, A. Aghababyan, L. Froehlich, O. Hensler, R. Kammering, K. Rehlich, V. Rybnikov,
Deutsches Elektronen-Synchrotron (DESY), Hamburg, Germany

Abstract

The linear, super-conducting accelerator at the new European XFEL facility will be able to produce up to 2700 electron bunches for each shot at a repetition rate of 10 Hz. The bunch repetition rate might vary initially between 100 kHz and 4.5 MHz to accommodate the various needs of experiments at three different SASE beam lines. A solution, which is able to provide bunch-resolved data of multiple data sources together in one place for each shot, has been implemented at the European XFEL as an integral part of the accelerator control system. This will serve as a framework for high-level control applications, including online monitoring and slow feedback services. A similar system has been successfully run at the FLASH facility at DESY for more than a decade now. This paper presents design, implementation and first experiences from commissioning the XFEL control system data acquisition.

INTRODUCTION

The idea of a shot-synchronized bunch-resolved data acquisition originates at DESY in the project for the superconducting TESLA test facility (TTF) and its successor, the Free-Electron Laser in Hamburg (FLASH). With TTF as a prototype for a superconducting linear accelerator and FLASH being the first free–electron laser linear accelerator it became imminent to record the data from beam diagnostics and RF devices for analysis purposes.

Since this type of linear accelerator is operated in a pulsed mode, it is desirable to collect the data at its pulse or shot repetition rate. This enables bunch-resolved studies of individual shot data. If all the data of interest for a given shot is made available in a single data structure, analysis will be quite simplified. Since this task is very similar compared to how high-energy physics experiments structure their data and perform subsequent processing and analysis on it, some of their concepts have been re-used to design an accelerator data acquisition type.

One of the conceptual key elements is the event record. It combines all data required for an analysis from various triggered data sources together in one data structure and stamps it with a unique event identification number. This event number is essentially the shot or pulse number of the linear accelerator RF pulse. The shot number attached to the data allows for an easy synchronization after acquisition and does not require the classic approach of retrieving parameters individually and then subsequently synchronize it via timestamps to make correlations e.g.

With the upcoming European XFEL project and its similarity to its smaller colleague FLASH, such a shot-synchronized and bunch-resolved data acquisition was chosen to be a central part of the accelerator control system. Running this kind of acquisition system for more than a decade at the FLASH facility provided a great deal of experience for designing and creating the new acquisition.

The following sections will briefly illustrate the design of the data acquisition system and its parts, present the implementation at the European XFEL linear accelerator control system and show some statistics of current operation.

DESIGN AND LAYOUT

The overall layout of the data acquisition as used for the European XFEL accelerator control system is shown in Fig. 1. It is based on the DOOCS control system framework [1].

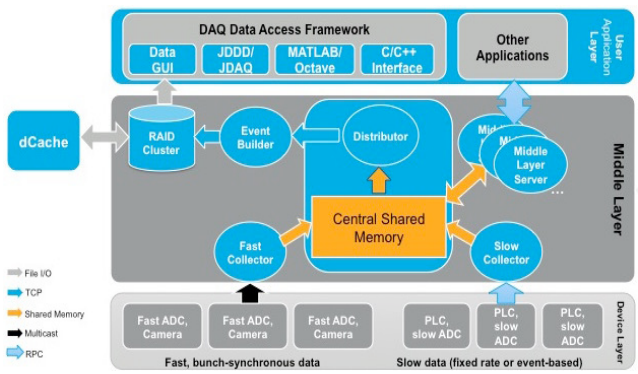

Figure 1: Schematic layout of a single data acquisition instance as used for acquiring shot-synchronized data from MicroTCA systems at the European XFEL linear accelerator control system.

On the device layer MicroTCA-based ADC modules, camera devices as well as PLC and other embedded devices are sending data to collector processes. A fast collector acquires data at the bunch repetition rate from triggered devices and a slow collector polling the data at rates of about 1 Hz from other hardware. The data has been stamped on the front-ends with a unique shot number provided by the timing system. Both collector processes are feeding the received data into a buffer manager using shared memory for storing the data. The distributor process functions as buffer manager and is in charge of managing the shared memory structure. Middle layer processes can connect to the buffer manager and read and/or

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUPHA210

write back to it. Once all data for a given shot number has been acquired in shared memory it is sent to the event builder and –writer processes. They will write the data as compressed files to disk. To tape it eventually the dCache [2] facility at DESY is being utilized. Several applications interfaces exist to read the data files and extract data for individual control system parameters (i.e. DOOCS channels on the front-end server). Middle layer server can provide data for other applications including graphical user interfaces for online monitoring purposes.

The following sections will explain the individual components of this design.

Front-End Devices

Providing a unique identification of individual pulses or shots is achieved by sending timing system information i.e. clock signals, trigger events etc. including a unique number created by the timing system master to all front-end devices. For the European XFEL accelerator control system this has been implemented using a timing system based on the MTCA.4 standard [3]. Since all front-ends for beam diagnostics, RF- and LLRF-controls are using the MicroTCA hardware platform a MTCA.4 module has been developed at DESY together with the University of Stockholm. This module is installed in every MicroTCA system. It receives the timing system information from the master via a fibre optic link at the shot repetition rate of 10 Hz. Clocks, event trigger and shot number are then provided to the MicroTCA system components via backplane as a PCIe interrupt or via M-LVDS lines. A timer server program on every MicroTCA system provides the timing system information also via ZeroMQ [4] to other applications as shown in Fig. 2.

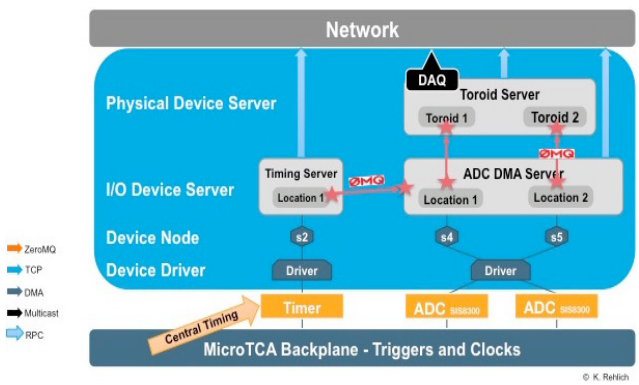

Figure 2: Standard read-out scheme for a beam diagnostics application on a MicroTCA system using the XFEL timing system to send data to the data acquisition system. The sender is represented here as the black box labeled DAQ.

The stamped data is sent via a push-type protocol based on Multicast UDP to all subscribers. That way multiple clients can connect to front-end devices without creating an additional load to the front-end systems. The instance on the front-end to subscribe to is called a "sender".

Collector

A collector process is running on a dedicated server node, usually a multi-core server with sufficient memory[1]. It subscribes to all front-end senders via Multicast, which are capable of acquiring data at bunch repetition rates. Hardware offering control system parameters of the accelerator control system without being connected to the XFEL timing system are retrieved via standard DOOCS RPC calls and stamped with the current shot number.

The data from an individual Multicast UDP sender is packed up into a so called "sender block" which combines all its DOOCS channels to be sent to the data acquisition system with a server block header. The header contains name, length, timestamp, number of channels, status and shot number to identify it. The collector allows for several retries until it would mark this specific server block as missing for this shot.

The collector process is registered as a client to the buffer manager and permitted to write to the shared memory area to store the server blocks.

The number of collector processes is not restricted. Several collector instances can connect to the same buffer manager. One individual collector process may have different locations for separating the received channels into smaller subsets.

Buffer Manager

The buffer manager [5] is the master of managing the shared memory area. It allocates the overall memory structure separated into a "client segment" for keeping all the client information subscribed to the buffer manager and an "event segment" for tracking the information about current shot data available in the shared memory. The data section itself holds all server blocks in memory for a couple of shots. These control and data segments are being set up according to a run control configuration. This configuration is aware of all sender locations of which data is to be expected and acquired.

Clients can subscribe for reading to all server blocks managed by the buffer manager or by specifying just a subset of what they are interested in. Clients can also write to the buffer manager like the collector processes do when filling the shared memory with received server blocks. Whenever expected server blocks for a given shot number are stored interested clients are being notified that the data is now available. DOOCS middle layer server with an interface to the buffer manager can subscribe to specific server blocks available in shared memory, too. Examples for this case are server processing shot-synchronous data of several front-end sender locations for displays e.g. the electron beam orbit derived from all BPM devices; or the energy measurement of the overall beam energy using LLRF data from all the RF stations. These middle layer servers can even write back the results of their computations to shared memory, e.g. the energy, and add it to the current shot data still in memory. The shot data is considered to be complete only if every one of

[1] Currently at minimum 128 GB up to 512 GB RAM is used for DAQ nodes.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUPHA210

the subscribed clients has been notified about the availability of the data. Every client must signal having received the data and all potential producers must have written back their data. Subsequently, the event record will be released from the shared memory after a waiting time. There are several mechanisms in place in case one of the clients is not present anymore, server blocks are missing or sender and clients are just very late in providing data. It is guaranteed that for every shot number an event record is created even if individual sender blocks are missing i.e. are potentially empty.

The distributor which hosts the buffer manager is furthermore able to group selected server blocks together to various event records and distribute them into data sets or so-called "streams". The configuration which server blocks form an event record and correspondingly a stream is defined by the run control configuration.

Event Builder and -writer

The event builder process connects to the distributor and receives the stream data via a dedicated TCP data link. This stream data is then distributed further to one event writer process per stream, which writes eventually the data files to disk. Data to be taped is written to the dCache facility. All other data is being kept for two days on online disks easily accessible by user nodes for analysis, then moved to an external disk space where it is stored for about one week.

Applications and Tools

To access the data file on disk several interfaces exist. There are API libraries available for C/C++, Java and MATLAB. A Python-enabled interface will be available soon. Several tools written in C/C++ and Java exist to just browse available data on disk and display it quickly e.g. the DAQ data GUI. Tools for extracting only a subset of the data help reducing the amount of data collected in a given file. An example of the DAQ data GUI is shown in Fig. 3.

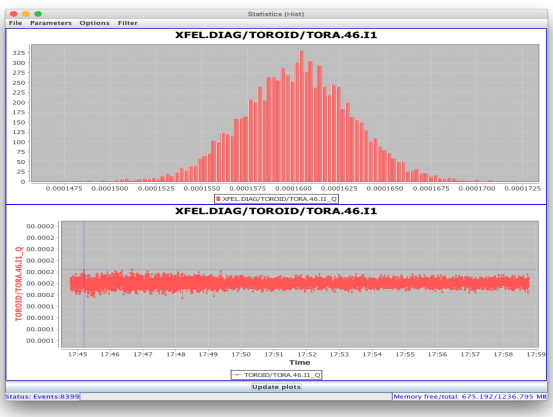

Figure 3: DAQ data GUI showing a histogram of a toroid charge channel and its time series.

The DOOCS-based JDDD client is able to read the data via a farm of data servers providing easy access to all data files. This is done by sending an XML-type request containing the time range for the desired data, the stream and the names of the DOOCS channels. In general, all above client interfaces can make use of data servers, too.

STATUS AND EXPERIENCES

The data acquisition system has been a central part of the conceptual design for the European XFEL accelerator control system from the beginning. The most important feature sought after is to provide shot-synchronous bunch-resolved data. But not only for archiving and later analysis but even more for online monitoring and computation of derived parameters relevant to accelerator operations. This had become evident in the 10 years of operation of several DOOCS data acquisition instances at the FLASH facility and the FLASH photon beam user experiments.

With two API implementations available to access the shared memory and hence the shot-synchronous data from a DOOCS middle layer server many applications have been meanwhile put into operation. One of the standard applications is to combine several front-end data sources to hand it over as one entity to the operation and status display. E.g. the orbit derived from the individual beam-position monitor data, the charge from individual toroid charge measurements, beam loss monitor data etc.

Higher-level physics applications e.g. beam energy measurement derived from the RF module data, beam power measurements, statistics on SASE production but also slow feedbacks have been meanwhile implemented as middle layer server. About a dozen server programs are running continuously on the main DAQ instance working in production mode. An example is shown in Fig. 4.

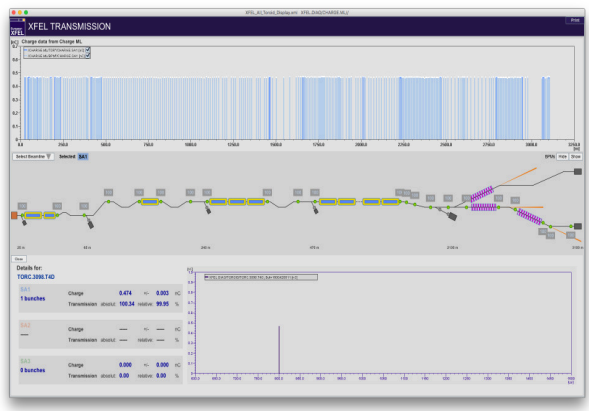

Figure 4: Transmission display for the European XFEL accelerator as derived from a DAQ middle layer server.

The first accelerator data acquisition instance has been taken into operation for the RF gun conditioning in December 2013. At the same time the Virtual XFEL was set up as a second DAQ instance. With the start of the injector operations and commissioning a third instance serving the RF- and LLRF-sender had been added. These instances already proved to be of value not only for beam operations of the injector part but also to analyze failure modes e.g. of individual cavities or RF modules. For commis-

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUPHA210

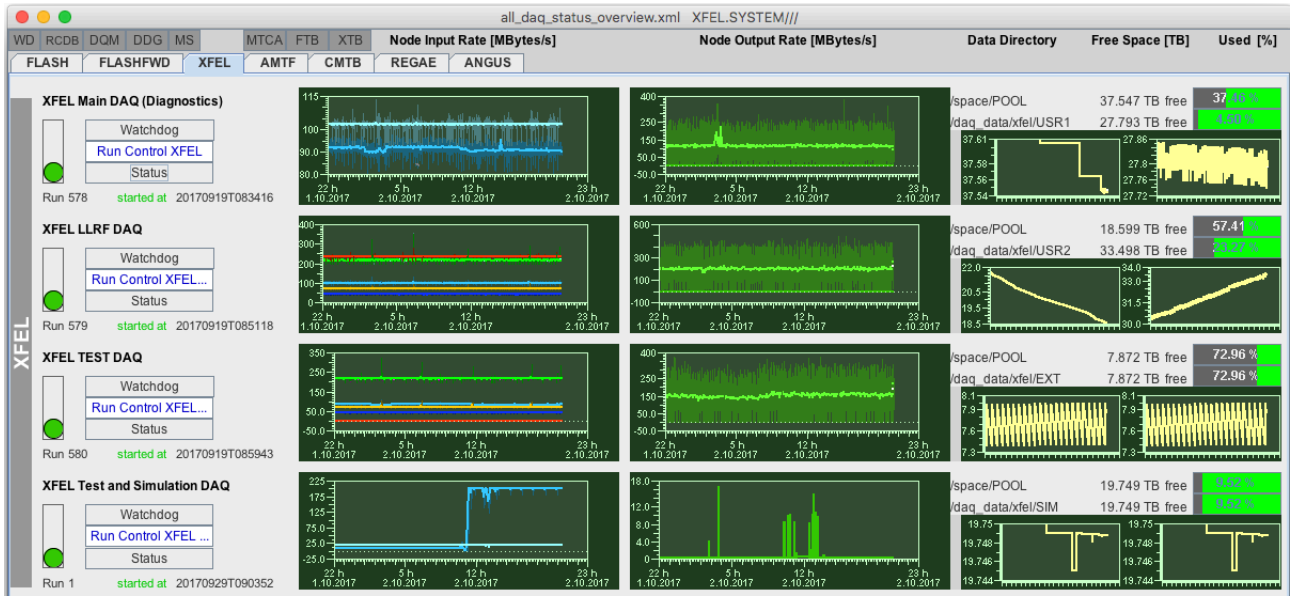

Figure 3: Overview of all accelerator data acquisition instances in operation for the European XFEL accelerator control system. Shown are input rates in the left green column of plots, output rates on the righ hand side.

sioning the full linear accelerator four instances in total had been set up. With the first user runs now in full swing all three "real" instances are operating 24/7.

Together all instances collect data at a sustained input rate of 1.5 GB/s from about 3.3 k of DAQ channels. These translate roughly into 13 k of DOOCS channels or control system parameters.

They can produce up to 30 TB per day of compressed data which is kept on local disk storage only for a few days but at least for one week on a dedicated storage area hosted in the DESY IT-Center.

CONCLUSION

The DOOCS-based shot-synchronous and bunch-resolved data acquisition system is a well-established solution, which has been proven to work quite well in the past for the FLASH accelerator and various FLASH photon beam user experiments. For the European XFEL accelerator control system this solution had been implemented from day one of the commissioning phase and so far demonstrated to be an essential and very useful part of the overall control system architecture as well as accelerator operations.

REFERENCES

[1] Distributed Object-Oriented Control System, DOOCS, http://doocs.desy.de/.
[2] dCache, http://dCache.desy.de/.
[3] H. Schlarb et al., "The case of MTCA.4: Managing the introduction of a new crate standard at large scale facilities and beyond", in *Proc. 14th Int. Conf. on Accelerator and Large Experimental Physics Control System (ICALEPCS'13)*, San Francisco, U.S.A., Oct. 2013, paper FMOPPC081
[4] ZeroMQ, http://zeromq.org/.
[5] V. Rybnikov et al., "Buffer manager implementation for the FLASH data acquisition", in *Proc. PCaPAC'08*, Ljubljana, Slovenia, Oct. 2008, paper TUP010