ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-WEPAB323

HIGH PERFORMANCE DAQ INFRASTRUCTURE TO ENABLE MACHINE LEARNING FOR THE ADVANCED PHOTON SOURCE UPGRADE*

G. Shen†, E. Chandler‡, N. Arnold, T. Berenc, J. Carwardine, T. Fors,
T. Madden, D. Paskvan, C. Roehrig, S. Shoaf, S. Veseli
Argonne National Laboratory, Lemont, USA

Abstract

It is well known that the efficiency of an advanced control algorithm like machine learning is as good as its data quality. Much recent progress in technology enables the massive data acquisition from a control system of modern particle accelerator, and the wide use of embedded controllers, like field-programmable gate arrays (FPGA), provides an opportunity to collect fast data from technical subsystems for monitoring, statistics, diagnostics or fault recording. To improve the data quality, at the APS Upgrade project, a general-purpose data acquisition (DAQ) system is under active development. The APS-U DAQ system collects a high-quality fast data from underneath embedded controllers, especially the FPGAs, with the manner of time-correlation and synchronously sampling, which could be used for commissioning, performance monitoring, troubleshooting and early fault detection, etc. This paper presents the design and latest progress of APS-U high performance DAQ infrastructure, as well as its preparation to enable the use of machine learning technology for APS-U, and its use cases at APS.

INTRODUCTION

Modern embedded controllers may contain several gigabytes of memory that are typically used for collecting fast data for statistics, diagnostics or fault recording. From a control system's point of view, collection, transfer and utilization of a large amount of data from numerous controllers represents a challenge that must be considered early on in the design cycle of a new accelerator.

The design of the APS-U Data Acquisition (DAQ) system was presented in [1, 2]. In this paper we provide brief update on progress with system implementation. In particular, we discuss development of a new set of DAQ simulation tools that enable testing software functionality and performance for various technical subsystems well before the actual hardware is in place. We also describe recent performance measurements done in order to ensure that APS network design can support specified data rates.

DAQ SYSTEM ARCHITECTURE

The main components of the DAQ system [1-3] are illustrated in Fig. 1. Wherever possible DAQ IOCs will directly interface with the technical system hardware where the acquisition is performed. In cases where direct connection to the technical system hardware is not feasible, a system specific "DAQ Front-end" will be used between the technical system and DAQ IOC. Each DAQ IOC will be somewhat customized for a given technical system but will also utilize a common framework for capturing and transferring time-series data. This framework will support the transmission of data across dedicated subnets, either directly to services for continuous data collection, or to services responsible for distribution of data to storage, external services, or applications prepared to accept and analyse this data.

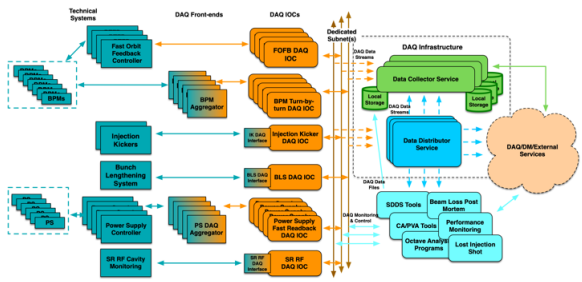

Figure 1: APS-U DAQ system main components.

Key aspects of the Data Acquisition (DAQ) System include:

- Capability to acquire time-correlated synchronously sampled data from several subsystems at various sample rates and correlate this data to within one beam revolution (3.6 µs) or better
- Most DAQ data includes a timestamp for each sample acquired allowing immediate plotting of data from various systems onto a common time-axis
- Support for continuous acquisition or triggered acquisition limited only by available storage
- The ability to route the data to any number of applications
- Use of EPICS PV Access [4-6] DAQ objects to encapsulate numerous signals to ensure data synchronicity
- Scalability by partitioning the heavy traffic on dedicated and multiple subnets and servers
- Separation from operational systems (networks, processors, servers) to allow troubleshooting & enhancements during user operations

The primary hardware components of the DAQ system are illustrated in Fig. 2. A server will be positioned at each double sector to receive and process the data from three nearby technical systems: Power Supply Fast Monitoring (PS) data, Fast Orbit Feedback (FOFB) data and BPM Turn-By-Turn (TBT) data. These double-sector servers isolate much of the "raw data" to local switches and pro-

* Argonne National Laboratory's work was supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences, under contract DE-AC02-06CH11357.
† gshen@anl.gov
‡ echandler@anl.gov

ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-WEPAB323

vide the first level of significant storage for storing long duration acquisitions. Other "one-of" DAQ systems (e.g. LFB, TFB, IEX, …) are connected where most convenient. Additional DAQ servers are in the main Computer Room. These "DAQ Central Servers" are used for data aggregation and user applications.

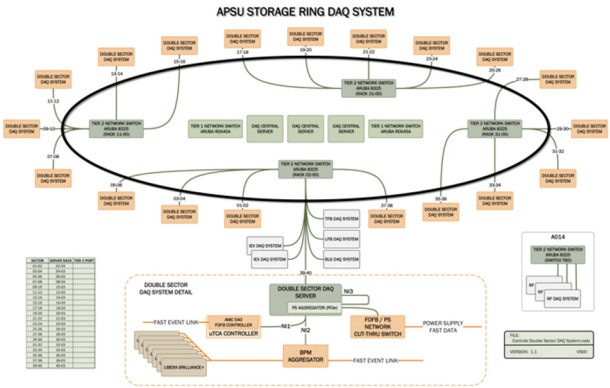

Figure 2: APS-U DAQ hardware architecture.

The main advantages of this approach are the common platform used for the three major DAQ subsystems, and also the cost-effectiveness of commodity hardware.

The components involved with data flow and processing of the DAQ data are numbered & identified in Fig. 3. These are the DAQ interface to the technical system (1), the acquisition and timestamping of the collected data (2), the DAQ IOCs (3), the communication medium and protocols between the IOCs and the services (4), the data collection and distributor services (5), and finally the client API and CLI interfaces for applications that consume the data (6). Also shown is the "processing stack" (7) that illustrates the layers of computing power available for data processing and manipulation. The following sections give additional details and suggested implementations of these components.

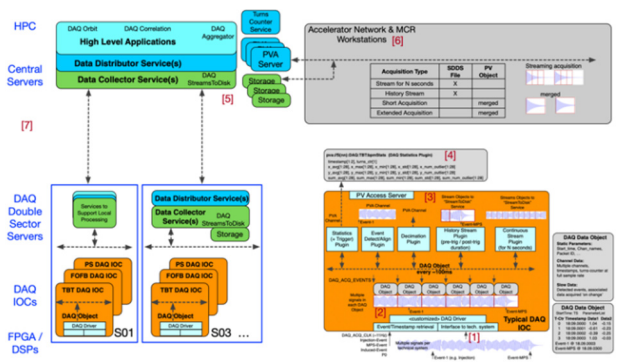

Figure 3: APS-U DAQ system data flow and processing.

As indicated in the bottom-right corner of the Fig. 3, the DAQ IOCs use Area Detector framework [7], which gives us a fair amount of flexibility in terms of processing plugins. The driver gathers data from the technical subsystem, assembles DAQ objects that are then passed through various processing and streaming plugins. The streaming plugins in particular are responsible for actually sending data directly to the data collector service. The history streaming plugins keep circular buffers of data and enable the system to collect data before and after a significant event happens. This can be incredibly useful tool for troubleshooting things like causes of beam dumps or other problems.

Note that all DAQ IOCs have a PVA server that is used to expose DAQ objects on the PV Access channel which can be retrieved using standard EPICS7 clients or APIs.

In addition to various high-level monitoring and processing applications, there are two DAQ services deployed in each double sector, which are 1) data collector service and 2) the distributor service respectively. The data collector service is responsible for storing data to local storage, while the distributor service forwards data to various client applications or other DAQ, data management, or external services.

IMPLEMENTATION PROGRESS

Over the last year we have made significant progress on the DAQ software implementation, especially in the areas of the common DAQ IOC and service infrastructure, simulation tools, and high-level applications. Some of the most notable IOC/service infrastructure achievements include the following:

- Generic UDP message protocol that will be utilized by most technical subsystems to send data to DAQ IOCs.
- Common IOC driver code with UDP listener and message processing framework.
- Common DAQ IOC Area Detector plugins used for capture and streaming of real-time and past data buffers to collector service, as well as for exposing raw data over EPICS PV Access protocol.
- DAQ IOC template modules that can be used for generating fully working IOC application, which requires minimal customizations that are relevant for a given technical subsystem.

As a result of these developments, we were able to significantly reduce DAQ base code size. This will ultimately result in increased speed of IOC deployment and simplify future maintenance effort. For example, we have already completed software development for BPM Turn-By-Turn, Power Supply Fast Monitoring, Booster Timing Controller, and Injection/Extraction DAQ IOCs.

In order to enable faster development and testing of different DAQ IOCs and high-level applications for monitoring and processing of DAQ data, we have implemented a set of simulation tools with the following features:

- Ability to generate system specific simulated DAQ data with different patterns and noise levels, at configurable sample and DAQ object rates, and to send this data to DAQ IOCs via UDP messages.
- Ability to drive DAQ IOCs via SDDS files at configurable sample and object rates.
- Ability to synchronize simulation for DAQ IOCs running at multiple double sectors.

We believe that these system simulation capabilities will be invaluable for development of various commissioning

ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-WEPAB323

scripts and other high-level DAQ tools. We have already prototyped some of the tools intended for monitoring, diagnostics, troubleshooting, and machine studies:

- DAQ Aggregator: utility that can be used to combine PV updates from multiple channels.
- DAQ Correlation and Alignment: set of tools for correlation and alignment of waveforms belonging to either to multiple PVA channels, or stored in different SDDS files
- DAQ Orbit: utility for collecting Turn-By-Turn BPM data and generating orbit waveforms.
- DAQ Turn Counter: utility for keeping track of beam turns and providing beam statistics.
- DAQ PV Group: tool that can be used to control and monitor multiple scalar CA PVs.
- Data Viewer: scope-like utility for viewing multiple channels of DAQ data in real-time.

Most of these utilities have ability to expose processed data as PV Access objects, as well as store this data into output SDDS files.

PERFORMANCE MEASUREMENTS

As mentioned above, there will be 20 double-sector DAQ servers distributed around the storage ring, each hosting the BPM TBT, PS and FOFB DAQ IOCs. Because of the numerous channels and fast acquisition, both the Turn-By-Turn DAQ and the Power Supply DAQ include FPGA-based "aggregators" which receive streams from the technical system hardware and repackage it for efficient processing in the DAQ IOC. Note that just those three double-sector subsystems taken together represent more than 13000 channels of data that can be collected simultaneously (see Table 1).

Table 1: Number of Channels and Data Rates for APS-U Double-Sector DAQ IOCs

Technical System DAQ	# of Channels	Sample Rate	Anticipated Data Rate (Per IOC)
Fast Orbit Feedback	256 (x 20)	22.6 kSPS	24MB/s
Power Supply Fast Monitoring	354 (x 20)	22.6 kSPS	32.3MB/s
BPM Turn-by-turn	84 (x 20)	271 kSPS	94MB/s

Several DAQ systems demand significant performance of the network and server components [1, 2]. Table 1 lists the anticipated data bandwidth for each double-sector DAQ system. The total double-sector data rate is about 150 MB/s, while the cumulative data rate for the entire DAQ system is over 3 GB/s. Note that generated Turn-By-Turn data could also include up to 13 additional signals per BPM, or up to 364 signals per double sector that were not counted in the Table 1 analysis. This could potentially result in almost 395 MB/s of additional TBT data per double sector, or almost 8 GB/s of data for the entire system.

We completed a series of performance tests in order to ensure that all IOCs running on a DAQ double-sector server can receive data from the technical systems at maximum rates. Designated server machine was able to easily handle the workload, and no single CPU was utilized over 60%.

CONCLUSIONS

APS-U DAQ is a time-correlated data acquisition system. Among other things, its main features are the ability to acquire data from multiple systems at different sample rates, support for continuous data acquisition, and ability to collect multiple signals within a single IOC. We made a fair amount of progress on implementing and enhancing DAQ software infrastructure, on development of system simulation capabilities, as well as on development of high-level tools.

We have already completed a number of networking tests in order to ensure that DAQ double-sector servers can receive and process data at maximum rates. We are currently performing a comprehensive set of system benchmarks for a fully deployed DAQ double-sector server with DAQ IOCs receiving data at a full capacity and serving this data to a number of monitoring clients via PVA Gateway.

We are also in the process of deploying the DAQ software and tools to an isolated HPC cluster to be used for DAQ system testing, development and testing of commissioning, diagnostics and monitoring tools, and running of simulations by accelerator physicists. One of our main future goals is to develop tools and models for machine learning based on DAQ data, and to utilize those tools for accelerator monitoring and diagnostics in real time.

ACKNOWLEDGEMENTS

The authors would like to thank all colleagues at APS facility and APS-U project for their fruitful discussions.

REFERENCES

[1] S. Veseli *et al.*, "Data Acquisition System for the APS Upgrade", in *Proc. ICALEPCS'19*, New York, USA, Oct. 2019, paper TUDPP02.

[2] G. Shen *et al.*, "High Performance Data Acquisition for a Modern Accelerator", in *Proc. IBIC'20*, Santos, Brazil, Sep. 2020, paper FRAO05.

[3] G. Shen, *et al.*, "High-Level Application Architecture Design for the Aps Upgrade", in *Proc. ICALEPCS'19*, New York, USA, Oct. 2019, paper WEPHA143.

[4] EPICS, `https://epics-controls.org`

[5] EPICS4 documentation, `http://epics-pvdata.sourceforge.net/literature.html`

[6] PVA specification, `http://epics-pvdata.sourceforge.net/pvAccess_Protocol_Specification.html`

[7] Area Detector documentation, `http://cars.uchicago.edu/software/epics/areaDetectorDoc.html`