ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA113

LINUX-BASED PXIe SYSTEM FOR THE REAL-TIME CONTROL OF NEW PAINTING BUMPER AT CERN

M. P. Pimentel*, C. Chanavat, E. Carlier, G. Gräwer, N. Magnin, N. Voumard, T. Gharsa
CERN, Geneva, Switzerland

Abstract

In the framework of the LHC Injectors Upgrade Project, the new connection from Linac4, injecting a 160 MeV H- beam into the Proton Synchrotron Booster (PSB) requires a set of four slow kicker magnets (KSW) per PSB ring to move the beam on a stripping foil, remove electrons and perform phase space painting. A new multiple-linear waveform generator based on a Marx topology powers each KSW, allowing adjustment of the current discharge shape with high flexibility for the different beam users.

To control these complex power generators, National Instruments (NI) PXIe crates fitted with a set of modules (A/D, D/A, FPGA, PROFINET) are used. Initially, control soft-ware developed with LabVIEW has validated the test bench hardware. A full software re-engineering, accessing the hard-ware using Linux drivers, C APIs and the C++ framework FESA3 under Linux CentOS7 was achieved for operational deployment.

This paper describes the hardware used, and the integration of NI PXIe systems into CERN controls environment, as well as the software architecture to access the hardware and provide PSB operators and kicker experts with the required control and supervision.

BACKGROUND

The PSB comprises four superposed rings, each equipped with four KSW magnets located at the injection area. A dedicated multi-waveform pulsed current generator individually powers each magnet [1, 2].

A single of those generators is composed of a master controller module, referred as KSW interlock card (KIC), and five capacitor charger/discharger (CCD) modules, 1x120V and 4x1200V. All of the mentioned modules are equipped with Anybus®-IC interfaces providing PROFINET communication interface [3]. To receive fast control commands, the generator power amplifier receives instructions by means of a hardwired connection with the FPGA controller board on the PXI crate. Finally, a feedback controller board receives an analog reference signal of the required waveform to be played in order to compare it with the discharge feedback and compensate any deviations perfecting the discharge shape. In addition to that, a pickup signal is available to read the final generators discharge.

A single Front-End Computer (FEC) controls and monitors all the mentioned peripherals on four generators corresponding to a ring. Fig. 1 is a simplified illustration of the described system.

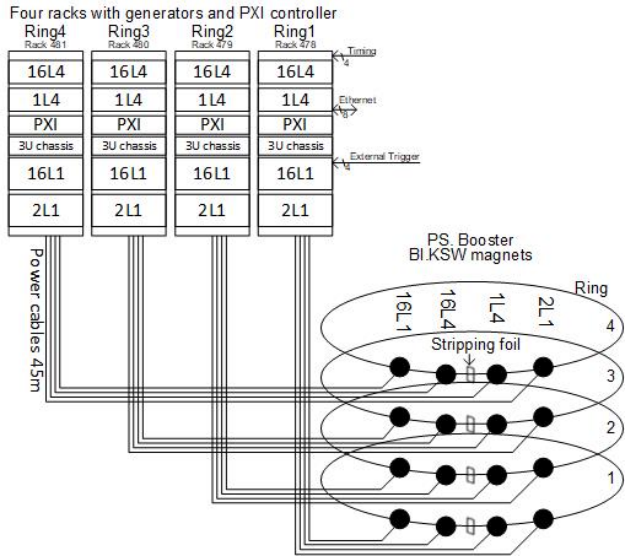

Figure 1: BI.KSW system.

CONTROL HARDWARE

The final FEC setup selected, to control four of those generators, consists in a NI PXIe-1082 chassis equipped with a PXIe-8821 controller. To interface with the PROFINET modules, the control crate is equipped with a Kunbus® DF-Profinet-IO operating as bus controller. The control crate hosts as well a field-programmable gate array (FPGA), NI PXIe-7841R, controlling four power amplifiers acting di-rectly in synchronisation with timing triggers. To ensure in-ternal processes synchronisation with the central timing, and generate part of the necessary timing synchronous triggers, a CERN custom hardware module, central timing receiver (CTR), incorporates the control crate. To generate the analog reference waveform, the crate also features an NI PXIe-6358 DAC module. In order to read the magnet discharge shape, a NI PXI-6132 digitizer is also incorporated.

OBJECTIVES

For early testing and validation of the prototype pulsed current generator, a test bench has been set up with all the components constituting a single ring. During this phase of prototyping and development, a first expert control system has been developed using LabVIEW™ as development environment.

In an effort to optimise the integration within the CERN accelerator control system and profit of all the existing generic tools, a Linux based approach has been retained for the final operational control system design. The software

* CERN TE-ABT-EC, marco.pimentel@cern.ch

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA113

re-engineering approach has been divided in three phases, as described bellow.

Phase One

The two main aspects requiring attention at an early stage for this approach are technological availability and performance validation against the strict system constraints.

Technological availability The first step was to install and test CERN CentOS 7 (CC7) Linux distribution and request the hardware manufacturers to provide the drivers and access libraries to evaluate their performance.

Performance validation The control system computes and re-configures thousands of parameters on all four generators every PSB cycle, i.e. every 1,2s. Taking into account the worst-case scenario charging time for a generator being 800ms. And considering the earliest event available in the timing, allowing the user identification, being 900ms before beam injection. The time frame available for the software to compute and configure all the electronics is 100ms. The bottleneck in terms of performance for this system was the PROFINET field bus communication. Iterations of tests and improvements in collaborations with the manufacturer lead to an acceptable speed of communication to configure all the 24 peripherals of a KSW setup.

Phase Two

After the system functionality, performance and stability validation, a system components analysis, separating system specific functionalities from wider technological functionalities, was performed aiming at reusability possibilities. From this analysis resulted the following structure, implemented in the form of C++ libraries:

- **Algorithm** (KSWAlgoLib) - Library regrouping all the required algorithms for computing operational commands and generating hardware settings.
- **Communication** (ProfiNetHWLib) - Library providing all PROFINET network configuration and communication, developed to be reusable in other applications.
- **Fast logic** (CNIRioLib) - Library interfacing with the project dependent gateware.
- **Waveform** (WaveGenCards) - Library providing interface with the DAC board, developed to be reusable in other applications.

Phase Three

In order to provide an operational API and integrate within CERN's control infrastructure and central timing, a final software layer is required, using CERN's FESA3 framework [6]. The created API consists of two intercommunicating classes, enabling a fully abstracted ring control from operation, and individual control over the generators by the experts. Additionally, a third existing generic class incorporates this architecture, to provide control over the equipment state. Aside from that, a separated process is running a generic class configured to control the CTR board and the generated interrupt signals synchronously with the accelerator timing. The pickup signals, caught by the data acquisition terminal, are in their turn captured by the internal post-operation check (IPOC) generic software. This application runs in its individual process [4].

SOFTWARE ARCHITECTURE

This paper will only address in more detail the part referring to the control and synchronization of the generators specifically developed for this project. Being the result of a reuse development, the IPOC system will not be deepened further. Figure 2 illustrates a simplistic overview of the control architecture software detailed in this document.

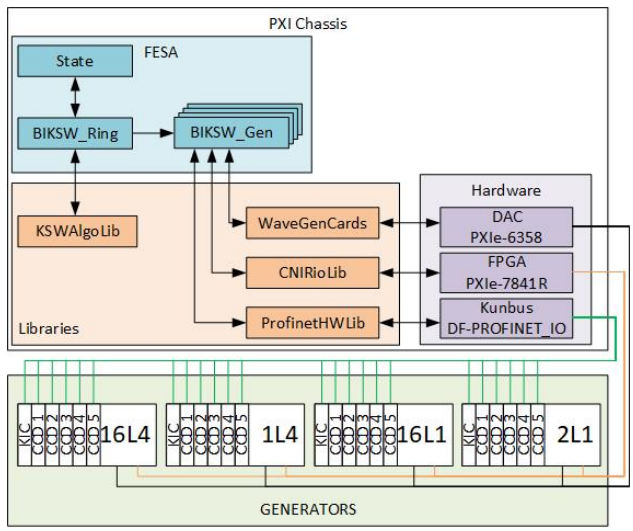

Figure 2: Generator control global architecture.

Algorithm Library (KSWAlgoLib)

This Library regroups all the algorithms for the validation and generation of the different waveforms [2]. Essentially, this library takes as an input the three first waveform coordinates. Each coordinate is defined by a time and amplitude, since the first point is always located at $t=0$. The operation is only required to input five values. Table 1 presents some metrics, illustrating the library dimension.

Table 1: KSWAlgoLib Metric Analyses

Metric	Value
Total Files	22
Total Lines	4957
Code Lines	3887
Functions Count	299
Classes Defined	25

According to specifications [2], there are four distinct types of waveforms, as illustrated by Fig. 3. Each of those waveforms require different configurations of the generators.

With this paradigm in mind, the architecture chosen for the implementation of this library is mainly based on the

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA113

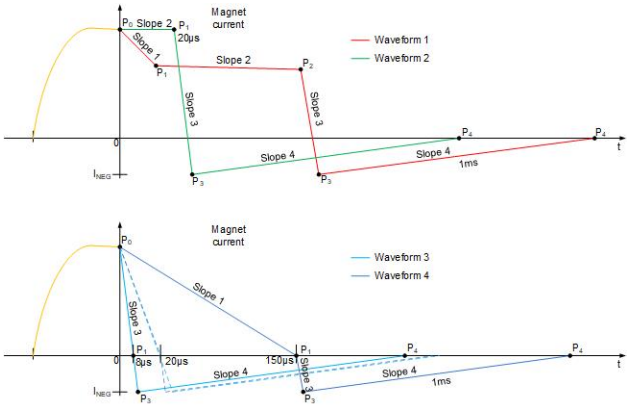

Figure 3: KSW possible discharge shapes.

chain of responsibility pattern [5]. The five received inputs are passed to a chain of handlers, each handler decides either to process the request or to pass it to the next handler in the chain until handled, or fall into an unmanageable situation raising a catchable error. This architecture provides great decoupling and expansion possibilities. In this library, a linear regression algorithm incorporating a "corners rounding" feature also found its place. Figure 4 illustrates a stripped down unified modelling language (UML) diagram of the final architecture of the library.

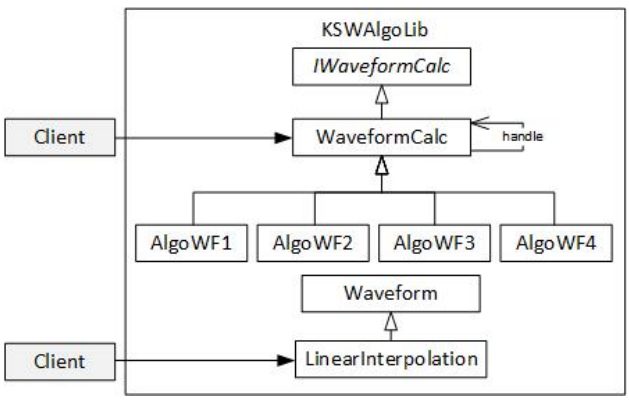

Figure 4: KSWAlgoLib Stripped Down UML.

Due to its high computational complexity and path diversity, this implementation is currently being submitted to more than 180 unitary tests, per waveform type, to validate its correctness. As shown in Fig. 4, clients can request computation data from "WaveformCalc" self-instantiable class. Taking the five operation points as input, this instance generates 30 configuration parameters and one 2×10 elements array per generator, or any bad parameter message if present.

This library can also take requests separately, to obtain the linear interpolation required for the DAC configuration. This second function is separated, because it is also required during expert operation with manually defined points, however, during remote operation its impute is the 2×10 array generated by "WaveformCalc" computation, returning a variable sized 2D array that can process at its maximum size 2×4064 elements for the higher defined sample rate of 3.125 Ms/s.

Communication Library (ProfiNetHWLib)

This particular piece of software takes care of the PROFINET communication management, not only by providing a communication interface, but also by preparing the network, locating and initialising all its components. This library includes implementations supporting the Kunbus® master controller board, and the in-house developed KIC and CCD 1200v and 120V with incorporated Anybus®-IC. Table 2 presents some metrics, illustrating the library dimension.

Table 2: ProfiNetHWLib Metric Analyses

Metric	Value
Total Files	32
Total Lines	3894
Code Lines	2641
Functions Count	197
Classes Defined	19

Due to this network diversity and elements multiplicity, the adopted architecture is based on the creational factory design pattern [5], and since it is a communication software, a notification mechanism is also incorporated to advertise new data in the network. Figure 5 illustrates the stripped down UML diagram of the library's architecture.

This architecture allows to instantiate any amount of devices of any of the proposed types dynamically. This implementation is engineered in such a way that all the instantiations are automatically constructed, by requesting to the master controller its generic station description (GSDML). Slaves are then mapped and the client only needs to call the instance pointers to start using any of the devices. This architecture also facilitates parallelism due to its notification mechanism. With this technique, each slave instance is responsible for the information exchange with the hardware at notification time, originating separated threads for each instance, greatly improving communication speed. This architecture provides a base easily extendable to add any new devices and controllers for other project. Additionally this library is prepared with a third supervisor category, not exposed in Fig. 5, allowing to add supervisory devices, like touch panels supporting PROFINET interfaces. One of the

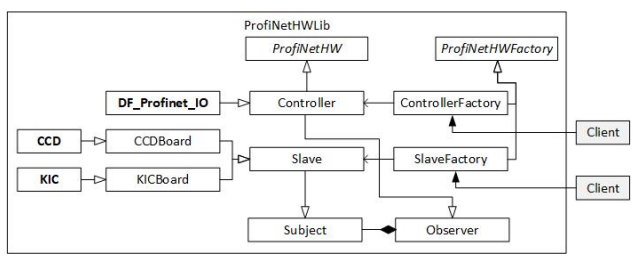

Figure 5: ProfiNetHWLib Stripped Down UML.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA113

particular challenges of this implementation, deserving mention, was the question of maintenance replacement of a CCD or KIC board. The PROFINET protocol ignores IPs for its configuration; only device names are used for the network lookup. As soon as the devices are found on the network, the IP addresses are overwritten according to the master configuration fi le. This fact requires every single replacement board to be programmed with the correct name prior to the replacement, demanding expert intervention. By running a sequence with pre-initialisation trough device control protocol (DCP) and simple network management protocol (SNMP), the problem has been solved, turning the whole PROFINET network into a plug-and-play system. Additionally, using the network supervisory functions, hot swap capability has been added (i. e. no system restart is needed to replace a CCD or KIC board).

Fast Logic Library (CNIRioLib)

The fast logic library is a C++ project specific library that simplifies the access to the NI RIO FPGA device from the KSW control software. Table 3 presents some metrics, illustrating the library dimension.

Table 3: CNIRioLib Metric Analyses

Metric	Value
Total Files	2
Total Lines	647
Code Lines	535
Functions Count	44
Classes Defined	11

It takes care of the driver initialization and the loading of the bitstream into the FPGA and it exposes business logic methods with structured parameters that will split/unsplit data to/from the FPGA registers. Being a project specific library, without reuse possibilities, there was no particular interest in designing a structured architecture for this software.

Waveform Library (WaveGenCards)

The WaveGenCards library defines an hardware abstraction layer, thanks to which it is easy to integrate new waveform generator card types without having to recompile the applications which will use it. The library API is based on a simple model of waveform generator functionalities, as shown in Fig. 6. According to this model, each waveform generator card has one horizontal control block, allowing the setting of the sample rate, the number of samples to generate, and the trigger source. Up to now only external trigger source is supported, with as parameter the input impedance, the level and the edge. Each waveform generator card contains a collection of output channels with configurable output impedance and coupling. Each channel provides a collection of ranges, every range having its own offset adjustment limits. The waveform to generate is set by output channel. The library also provides the horizontal characteristics for each waveform generator card, as well as the vertical characteristics for each of its channels, describing the possible values for all the parameters of a given waveform generator card type.

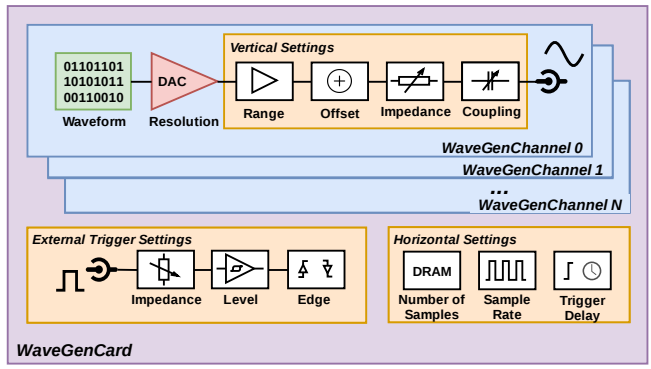

Figure 6: Simple waveform generator card model.

Table 4 presents some metrics, illustrating the library dimension.

Table 4: WaveGenCards Metric Analyses

Metric	Value
Total Files	51
Total Lines	6987
Code Lines	4833
Functions Count	279
Classes Defined	38

Real-Time Software

The Front-End Software Architecture, known as FESA, is a complete environment for the equipment specialists to design, develop, deploy and test their equipment software, called a FESA class [6]. These C++ generated FESA classes provide a base for control integration with CERN's infrastructure and services. Two classes have been developed, in this project context, using this framework.

Generator Class One instance of this class is deployed per generator, giving access to all its detailed configuration and acquisitions parameters, this class is protected by a role-based access control (RBAC) and is only accessible by machine experts for intervention purpose. This class also rejects any command if the equipment is running in remote mode. There are two distinct groups of data in a FESA class, being global data and instance data. Global data refers to data shared between all running instances, while instance data is particular to a single instance. Table 5 gives metrics about this class and the non-generated files implemented in it.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOPHA113

Table 5: Generator Metric Analyses

Metric	Value
Global settings fields	8
Global Acquisition Fields	23
Instance Setting Fields	42
Instance Acquisition Fields	162
Total Files	10
Total Lines	3167
Code Lines	1099
Functions Count	72
Classes Defined	6

Ring Class Only one class instance comes on top of the four generator instances abstracting all their complexity into only 6 parameters to the operation, 5 of them the definition of the 3 first points of a discharge curve, and one kick enable or disable option. All these options are Pulse-to-Pulse modulation settings. This class also processes other settings and acquisitions. Reserved to operation experts, there are parameters defining the nominal amplitude and operational limits. Exclusively for the equipment experts, computation constants and equipment limits can be defined. Therefore three sets of RBAC authorisations are defined for different sets of controls. Table 6 gives metrics about this class and the non-generated files implemented in it.

Table 6: Ring Metric Analyses

Metric	Value
Global settings fields	26
Global Acquisition Fields	0
Instance Setting Fields	22
Instance Acquisition Fields	8
Total Files	12
Total Lines	143
Code Lines	89
Functions Count	6
Classes Defined	1

CONCLUSION AND OUTLOOK

Through this project, the use of non-proprietary operating system and software has been successfully validated on a PXI platform using only off-the-shelf modules. Nevertheless, the use of a proprietary software solution at the early stage of the project has permitted to validate the different power components with at the same time an easiest and flexible approach to implement, test, validate and modify the required control functionalities. The re-engineering of the available proprietary based software within the CERN accelerator control system has permitted to optimise the software structure while keeping the available hardware. This approach opens the door for a reduction on the dependencies from proprietary software, while homogenising the control software across different type of equipment.

The results obtained on the test bench implementing a full vertical slice of the control system shown a great performance, with an average of 56ms, to compute and configure all the hardware for a ring.

The second valuable aspect of this project is its organisation around generic libraries for PROFINET communication and waveform generation providing a reusable base for future projects.

REFERENCES

[1] L. Feliciano *et al.*, "A new hardware design for PSB kicker magnets (KSW) for the 35 mm transverse painting in the horizontal plane", in *Proc. 6th International Particle Accelerator Conference (IPAC'15)*, Richmond, VA, USA, 3 - 8 May 2015, pp. 3890–3892. doi:10.18429/JACoW-IPAC2015-THPF086

[2] G. Grawer *et al.*, "A multi-waveform pulsed current generator for slow kicker magnets", in *Proc. 2018 IEEE Power Modulator and High Voltage Conference*, Jackson Lake Lodge, USA, 3-7 June 2018, pp.105, https://indi.to/5LTqJ

[3] N. Voumard *et al.*, "Use of multi-network fieldbus for integration of low-level intelligent controller within control architecture of fast pulsed system at CERN", presented at the 17th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'19), New York, NY, USA, Oct. 2019, paper MOPHA152, this conference.

[4] N. Magnin *et al.*, "Internal post operation check system for kicker magnet current waveforms surveillance", in *Proc. 14th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'13)*, San Francisco, CA, USA, 6 - 11 Oct 2013, paper MOPPC029, pp. 131–134, https://cds.cern.ch/record/1696972

[5] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *"Design Patterns: Elements of Reusable Object-Oriented Software"*, Addison-Wesley, 1995.

[6] M. Arruat *et al.*, "Front-end software architecture", in *Proc. 11th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'07)*, Knoxville, TN, USA, 15 - 19 Oct 2007, paper WOPA04, https://accelconf.web.cern.ch/accelconf/ica07/PAPERS/WOPA04.PDF