ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THPDP080

GATEWARE AND SOFTWARE FOR ALS-U INSTRUMENTATION *

L. M. Russo†, A. Amodio, M. J. Chin, W. E. Norum, K. Penney, G. J. Portmann, J. M. Weber
LBNL, Berkeley, USA

Abstract

The Advanced Light Source Upgrade (ALS-U) is a diffraction-limited light source upgrade project under development at the Lawrence Berkeley National Laboratory. The Instrumentation team is responsible for developing hardware, gateware, embedded software and control system integration for diagnostics projects, including Beam Position Monitor (BPM), Fast Orbit Feedback (FOFB), High Speed Digitizer (HSD), Beam Current Monitor (BCM), as well as Fast Machine Protection System (FMPS) and Timing. This paper describes the gateware and software approach to these projects, its challenges, tests and integration plans for the novel accumulation and storage rings and transfer lines.

INTRODUCTION

The ALS-U project is currently under development at the Lawrence Berkeley National Laboratory, in Berkeley. It consists of a new 2 GeV Storage Ring (SR), a new full-energy Accumulation Ring (AR) and new Transfer Lines between the SR and AR, while reusing the existing Injector [1].

The Instrumentation team was tasked with upgrading the systems for which it's responsible, while maintaining the network-attached device (NAD) architecture currently in use at ALS [2]. Moreover, because all systems, with the exception of the FMPS, have been in operation at ALS for years, it's important to minimize operation disruption, while upgrading the designs in an isolated and uniform way.

HARDWARE

The Instrumentation projects maximize the usage of open-source hardware solutions in a way to improve robustness against supply chain issues, component obsolescence and collaborative nature. To that end, a dual FMC FPGA carrier board called Marble [3], developed internally and released under the CERN Open Hardware License v1.2 [4], was selected for multiple systems: Fast Orbit Feedback, Fast Machine Protection System, Event Generator, Event Fanout and Event Receiver. The board features a Xilinx Kintex-7 FPGA, DDR3 SODIMM, dual FMC slots, STM32 microcontroller for board management and a flexible clocking scheme. The board can be seen in Fig. 1.

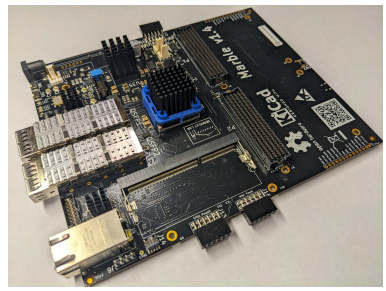

Figure 1: Marble v1.4.

For systems that require faster ADCs, such as the HSD, BCM and BPM, the Xilinx RFSoC SoC [5] family achieved all the requirements. Two demoboards featuring this part were selected: ZCU111 [6], featuring the Gen2 ZU28DR device, consisting of 8 12-bit 4 GSPS ADCs, 8 14-bit 6.5 GSPS DACs, and a quad-core ARM A53 processor; ZCU208 [7], featuring the Gen3 ZU47DR/ZU48DR devices, consisting of 8 14-bit 5 GSPS ADCs, 8 14-bit 9.85 GSPS DACs, and a quad-core ARM A53 processor. The boards can be seen in Fig. 2.

Figure 2: Xilinx ZCU111 (left) and ZCU208 (right).

COMMON CODEBASE AND CONTINUOUS INTEGRATION

All Instrumentation FPGA projects are being developed using a common, reproducible and extensible build system based on Makefiles and TCL scripts from a repository called Bedrock [8]. It includes, among rules for generating test-benches and synthesis/implementation, a variety of portable Verilog modules, ranging from common digital logic designs like FIFOs, RAMs and pulse synchronizers, to more complex operations like multi-gigabit fiber protocols and DSP operations. Also, Bedrock is based on free, open-source and well-known tools and languages to reduce external dependencies, while maintaining long-term support and reducing the barrier for collaboration.

As Bedrock, Instrumentation repositories are following Continuous Integration practices, by leveraging Gitlab [9] and Gitlab runner [10] projects to provide automatic testing and a valid final bitstream at each commit. There are also plans to support Hardware-in-the-loop testing [11].

* Work supported by the Director, Office of Science, Office of Basic Energy Sciences, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.
† lucasrusso@lbl.gov

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THPDP080

BEAM POSITION MONITOR

The Beam Position Monitor (BPM) is based on the Xilinx Gen3 RFSoC ZU47DR/ZU48DR devices. The hardware platform chosen was the Xilinx demoboard ZCU208 due to its excellent testbench results. The gateware and embedded software is based on the existing ALS BPM [12], specifically the 500 MHz AM demodulation digital signal processing chain, pilot tone calibration and data acquisition. However, the ADC data capture logic and calibration, as well as the DAC control for generating the 500 MHz pilot tones, and board control had to be redesigned or developed.

In order to leverage the high-speed (5GSPS) 14-bit ADCs present on this SoC, while reusing the existing BPM design, the downconversion (DDC) stage was split into two. The first one is done inside the ADC block itself (RF Data Converter or RFDC) and the second stage is done entirely in FPGA fabric.

The ADC sampling clock is generated by an external module, called CLK 104 [13] and its frequency was chosen in order to maximize the sampling speed while respecting the rational relationship between the machine *RF*, harmonic number (h), first stage decimaton rate (D) and an arbitrary number N. The *RF* and h have a special meaning in which $\frac{RF}{h}$ is known as the storage ring orbit clock (*SROC*) and the number N can be interpreted as the number of DDC samples desired in one *SROC* period.

$$ADC_{freq} = \frac{RF * N}{h} * D = \frac{500 * 81}{328} * 40 \quad (1)$$

$$ADC_{freq} = 4.939\,\text{GSPS} \quad (2)$$

In addition to that, the sampling clock must be synchronized with the machine master oscillator to provide accurate measurement. Because of that, and to respect the relationship derived in Eq. (1), the local crystal oscillator (i.e., VCXO) frequency must be harmonically related to *SROC*.

The sampling clock generated by the CLK104 board is routed directly to dedicated pins in the RFSoC and used by the RFDC block. Apart from this, two more sets of clocks are generated and used by the FPGA design. The first ones are called "ADC refclk" and "DAC refclk" and are used to capture/send downconverted or baseband data from/to the RFDC block. The second one is called "PL sysref", and is used to synchronize different RFDC ADC/DAC blocks, in a way that the same sample across multiple ADCs/DACs refers to the same high-speed clock period.

The first ADC conversion stage consists of a differential to single to single-ended module plus a DDC, shifting the RF from 500 MHz to 30 MHz and reducing the sampling speed from 4.939 GSPS to 123.48 MSPS. For the Pilot Tone generation, a DAC will be used to used to generate the above and below RF calibration tones. It consists of a parallel to serial data interface and an upconversion stage. This is shown in Fig. 3.

After the first demodulation stage, the complex downconverted ADC samples are fed into a complex mixer tuned to shift the signal of interest to baseband, as shown in Fig. 4. The RF products are the result of this operation and used for the final stage.

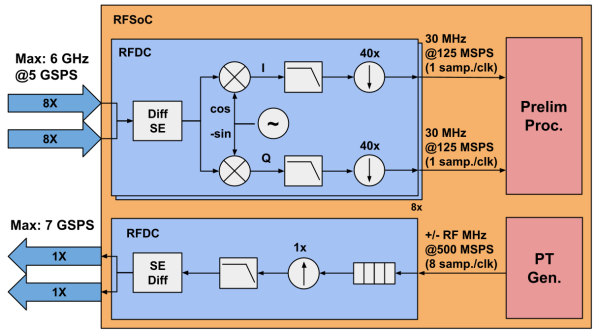

Figure 3: BPM RFDC downconversion/upconversion stage.

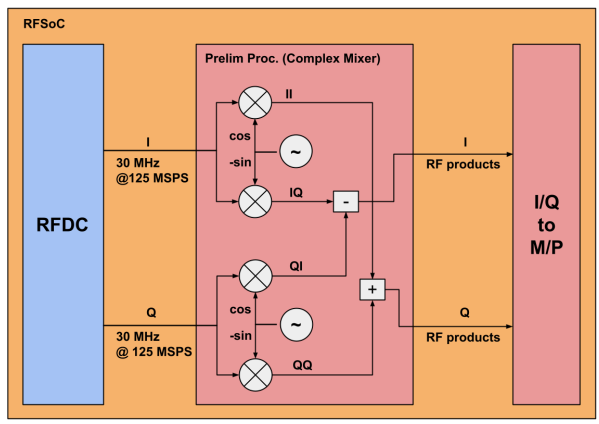

Figure 4: BPM complex mixer.

This final stage consists of 2 different low-pass filters, selected to output a Turn-by-turn data rate and an integer number of turns (dependant on the Pilot Tone parameters, as described in [12]). The two data streams are then converted from I/Q coordinates to amplitude/phase by a CORDIC and, finally, decimated to the Fast Acquisition and Slow Acquisition data streams used by the Fast Orbit Feedback system and for monitoring purposes, respectively, as shown in Fig. 5. The Pilot Tone calibration continuously adjust the output amplitude gains, in order to achieve, primarily, long term stability.

The remainder of the gateware/software includes the data acquisition engine, calibration routines, embedded timing interface, analog front end control and embedded software for status monitoring and EPICS interface. The code is open-source and available on Github [14].

FAST ORBIT FEEDBACK

The Fast Orbit Feedback (FOFB) is based on the same architecture as ALS, consisting of a distributed 12-node (one per sector) data concentrator organized in a bidirectional ring topology. The data concentrator, called Cell Controller, receives position data in two ways. The first one, from local

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THPDP080

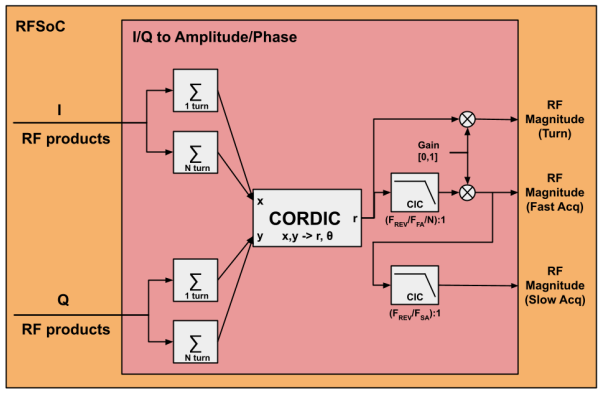

Figure 5: BPM I/Q to Amplitude/Phase.

sector BPMs, via 2 unidirectional MGT links forming a local ring. The second one, from other sector BPMs, via 2 bidirectional MGT links connected to neighbor Cell Controllers, forming a larger ring. During each FOFB cycle, each Cell Controller forwards or discards (FoD) a packet based on a received packet bitmap. Finally, each concentrator is connected to the sector Power Supply Controllers (PSC) via an Ethernet link in a daisy-chain fashion, forming a sector PSC ring, by which it sends new digital setpoints and receives readbacks [2].

The fundamental difference between the FOFB system in operation at ALS and the ALS-U FOFB system is a hardware platform change. The former uses a closed-source hardware FPGA carrier board called BMB7 [15], making it difficult to manufacture and use in new systems in light of obsolete components, while the latter is the open-source hardware design called Marble. Because the two platforms adopt different approaches to network communication, clock distribution and board management, adaptations are necessary for both the gateware and embedded software. No changes in the algorithm are necessary.

The data distribution algorithm is done in two parts. The first one is composed of a local sector BPM ring, in which local sector BPMs data are read. As shown in Fig. 6, two unidirectional MGT links are used to receive data. The Aurora streams are merged by an AXI-Stream Interconnect [16] and a bitmap is used to keep track of the data received.

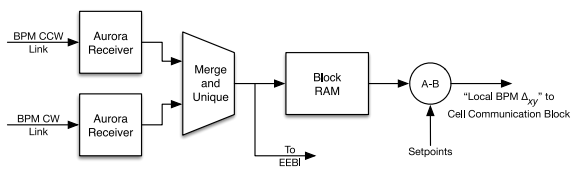

Figure 6: FOFB local BPM Forward or Discard.

Each BPM generates a packet such as the one shown in Fig. 7, which is composed of a header uniquely identifing the packet in the network, BPM position data X and Y along with the Sum signal.

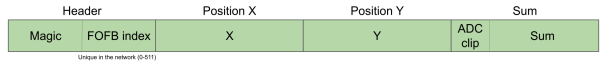

Figure 7: FOFB BPM packet.

The Cell Controller, on receiving the BPM packet modifies it to include the Cell Controller index in the header and to subtract the position offset received from the control system. This can be seen in Fig. 8.

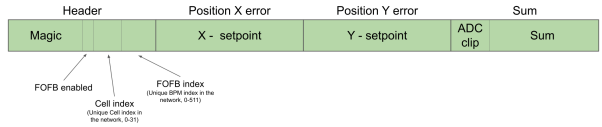

Figure 8: FOFB Cell Controller packet.

The second part of the algorithm is composed of a Cell Controller ring, in which neighbor Cell Controllers send/receive BPM data to/from other sectors. In the same way as the first part, AXI-Stream Interconnect blocks are used to merge the streams and a global bitmap is used to keep track of all data received by all local and remote BPMs. The BPM data, in turn, is sent to a block RAM to be used by the FOFB DSP. This can be seen in Fig. 9.

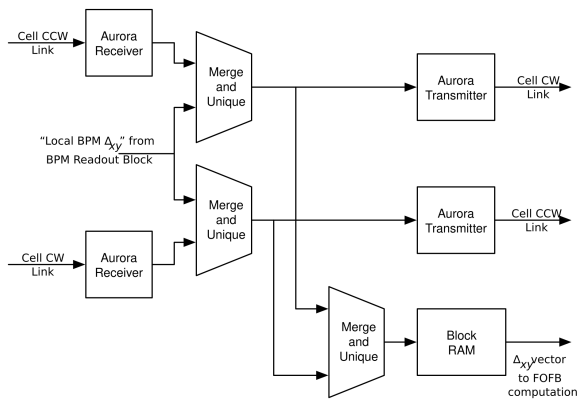

Figure 9: FOFB Cell Controller Forward or Discard.

The code is available on [17].

HIGH SPEED DIGITIZER AND BUNCH CURRENT MONITOR

The High Speed Digitizer (HSD) and Bunch Current Monitor (BCM) [18] have already been developed for ALS and is based on the Xilinx Gen2 RFSoC demoboard ZCU111. In order to leverage more modern RFSoC devices new HSD/BCM units could be manufactured using the same device as the BPM, the ZCU208, for which the gateware/software adaptations have already been made. In this case, a possible analog front-end redesign is needed, due to component obsolescence.

Both the HSD and BCM are meant to mimic the functionality of a real-time and sampling oscilloscope, respectively. They both share the same codebase differing only by the

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THPDP080

acquisition engine. The HSD ADCs run at a sampling rate of $RF * 8 = 4GHz$, with the RFDC block generating 8 samples per AXI-Stream clock cycle (500 MHz) into a set of BRAMs bundled together to provide 8 channels, 12-bit, 128k samples. On the other hand, the BCM ADCs run at a sampling rate of $RF * 80/11 = 3.6GHz$ and the acquisition engine stores the sum at each point over multiple turns.

The overall top-level diagram gateware can be seen in Fig. 10. It consists of the following blocks: RFDC module, sourcing the raw ADC data via AXI-Stream protocol; an embedded EVR module, which recovers an RF/4 clock, events and distributed bus from a timing link; a hard-core ARM core, responsible for housekeeping tasks, board control/status and EPICS interface; an acquisition module, which captures either the raw ADC data directly into BRAM or a sum at each point, similar to a sampling oscilloscope.

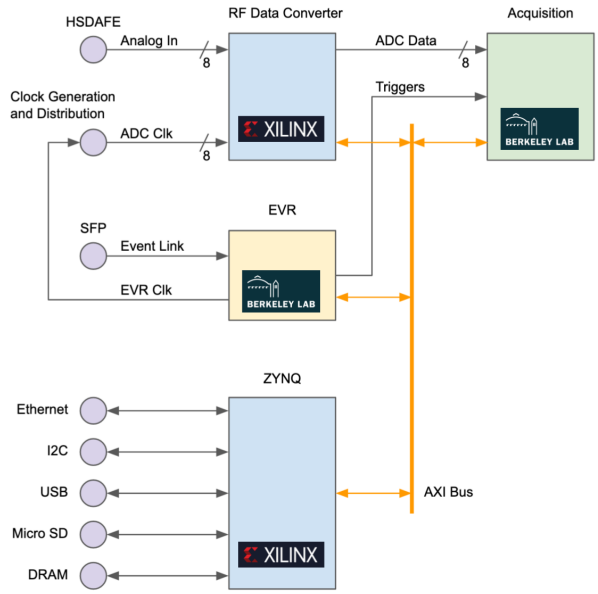

Figure 10: HSD/BCM top-level diagram.

More information can be found in [18] and all the code is available on [19].

FAST MACHINE PROTECTION SYSTEM

The Fast Machine Protection System (FMPS) leverages the FOFB network, as well as Cell Controller hardware to collect local sector interlock signals, to transmit MPS signals to all sectors. In addition, the FMPS consists of a separate node, called Mitigation node, responsible for calculating the MPS algorithm and sending signals to the RF system. This node acts like a 13th node in the FOFB network, forwarding or discarding packets like a regular Cell Controller.

The FMPS packet, shown in Fig. 11, contains a header, used to uniquely identify its origin and status bits, which can contain 2 types of information: (i) status from the each sector, such as fast valves and cold-cathode gauges, sent to the Mitigation node; status from the RF system, such as beam status and permit/heartbeat, sent back to each sector.

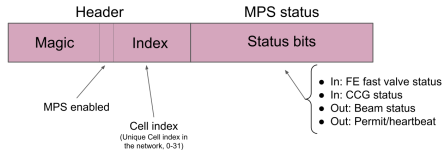

Figure 11: FMPS packet.

Overall, from the FOFB network point of view, each Cell Controller generates BPM data, collected from each local BPM, and FMPS data, collected from local gauges and valves. Both types of data follow the same FoD algorithm for received data, but differ on its consumers: Cell Controllers consume only FOFB packets to calculate the FOFB algorithm; the Mitigation node consumes both FOFB and FMPS data to determine what signals need to be sent to the RF system.

Conversely, the Mitigation node generates a special FMPS packet containing status information back to each sector and it's not time sensitive. The Cell Controllers differentiate this FMPS packet from others by a special header information.

TIMING SYSTEM

The Timing system for ALS-U consists of an in-house designed hardware, gateware and embedded software for all components: Event Generator, Event Fanout and Event Receiver. It is compatible with commercial solutions [20], used by many light sources, and uses Marble as a hardware platform with in-house designed FMC modules to fan-out timing links, called FMC EVIO, or to generate triggers, patterns, clocks synchronized with the master oscillator, called FMC EVRIO [2].

One of the main requirements of the ALS-U timing system is the injection across 2 different RF domains (RF_{inj} and RF_{sr}), with the rational relationship $RF_{inj} = \frac{608}{609} * RF_{sr}$: (i) injector (Linac and Booster) domain $RF_{inj} = 499.57MHz$ and (ii) AR/SR domain $RF_{sr} = 500.39MHz$. This requirement triggered the development of a custom Event Generator supporting 2 frequency domains.

Each domain is clocked by an external reference clock, coming from the Master Oscillator Distribution system. These clocks are used as reference clocks to the FPGA Multi-Gigabit Transceivers (MGT) Channel Phase-Locked Loop (CPLL) which, in turn, generate the transmitter clocks. In order to properly synchronize the 2 domains each domain needs to detect the coincidence between the other domain clocks in respect to its own reference clock. This is done according to Fig. 12.

The reference and transmitter clocks from one domain are sampled by the reference clock of the other domain and an histogram is recorded. The number of bins in the histogram is defined by the number of sample clock cycles in the coincidence interval, which in this case is either 608 or 609. With the histogram captured, the software sweeps the results looking for the rising edge of the sampled clock. This infor-

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THPDP080

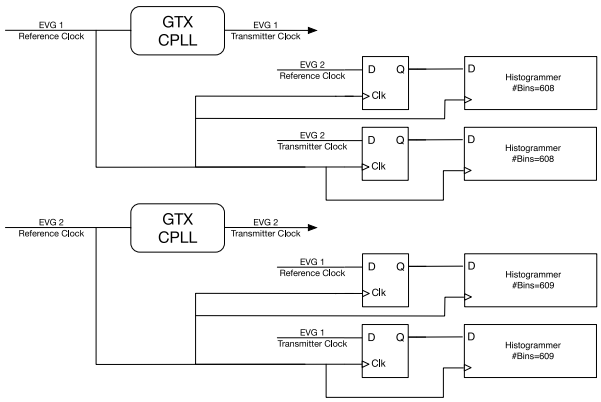

Figure 12: Dual Event Generator coincidence detection.

mation is then used to properly generate a heartbeat event used to synchronize the two domains.

Moreover, the Dual EVG is flexible and supports the generation of software-triggered events, hardware-triggered events and sequence events, over the event stream, and data, over the distributed data bus.

The Event Fanout/Receiver nodes shares the same codebase and differs only by the choice of the FMC module, which can be the FMC EVIO, for the Event Fanout, or the FMC EVRIO, for the Event Receiver. The Fanout replicates the input stream to every output without the need to go through the FPGA, by relying on a 40x40 crosspoint switch on the FMC module. In this case, the FPGA has an embedded EVR module only for monitoring purposes. On the other hand, the Receiver decodes the timing link and generates synchronized electrical triggers, patterns and clocks, by using the PLLs and buffers available on the FMC EVRIO.

The Dual EVG code is available on [21] and the Event Fanout/Receiver is available on [22].

FUTURE WORK

The systems described in this paper are in various degree of completion, with the Timing and HSD/BCM being the most advanced in terms of gateware/software, control system integration and field tests, even though a new sequencer for the Dual EVG will be needed for the new AR injection. The BPM is also well advanced, with tests already being done in the field, with good results. A lot of work has been done to add/improve simulation capabilities for the FOFB codebase and also to support the new hardware platform with the initial tests with the hardware show promising results. FMPS progress will heavily rely on the FOFB codebase initially to provide the network needed for FMPS. As such, the next goal is to start developing the FMPS Mitigation node networking interface, which will reuse most of the FOFB code, and the interlock algorithm, which has to be written from scratch.

CONCLUSION

ALS-U Instrumentation scope encompasses multiple critical projects, such as Timing, BPM and FMPS, of which if not working properly could directly impact ALS-U uptime. A lot of work has been done to reuse existing and tested code from ALS, but there are still adaptations and development needed to support all of the necessary features and to meet the new AR/SR requirements. Nevertheless, the demonstrated results are satisfactory and the increasing use of CI, along with simulation and hardware-in-the-loop tests, reduce the chance of regressions while improving the overall confidence on the codebase.

REFERENCES

[1] C. Steier, Ph. Amstutz, K.M. Baptiste, P.A. Bong, E.S. Buice, P.W. Casey, *et al.*, "Design Progress of ALS-U, the Soft X-ray Diffraction Limited Upgrade of the Advanced Light Source", in *Proc. IPAC'19*, Melbourne, Australia, May 2019, pp. 1639–1642, doi:10.18429/JACoW-IPAC2019-TUPGW097

[2] J. M. Weber, J. C. Bell, M. J. Chin, S. De Santis, R. F. Gunion, W. K. Lewis, *et al.*, "ALS-U Instrumentation Overview", in *Proc. IPAC'21*, Campinas, SP, Brazil, May 2021, pp. 3427–3429. doi:10.18429/JACoW-IPAC2021-WEPAB321

[3] LBNL FPGA Carrier Board, Marble, https://github.com/BerkeleyLab/Marble

[4] CERN Open Hardware License v1.2, https://ohwr.org/project/licences/wikis/cern-ohl-v1.2

[5] Xilinx Zynq Ultrascale+ RFSoC techonology, https://www.xilinx.com/products/silicon-devices/soc/rfsoc.html

[6] Zynq UltraScale+ RFSoC ZCU111 Evaluation Kit, https://www.xilinx.com/products/boards-and-kits/zcu111.html

[7] Zynq UltraScale+ RFSoC ZCU208 Evaluation Kit, https://www.xilinx.com/products/boards-and-kits/zcu208.html

[8] LBNL Bedrock Common Repository, https://github.com/BerkeleyLab/Bedrock

[9] Gitlab, https://about.gitlab.com

[10] Gitlab Runner, https://docs.gitlab.com/runner

[11] C. Serrano, M. Betz, L.R. Doolittle, S. Paiagua, and V.K. Vytla, "Hardware-in-the-Loop Testing of Accelerator Firmware", in *Proc. ICALEPCS'19*, New York, NY, USA, Oct. 2019, pp. 659–664. doi:10.18429/JACoW-ICALEPCS2019-TUAPP01

[12] G.J. Portmann, M.J. Chin, W.E. Norum, and J.M. Weber, "BPM Electronics With Self-Calibration at the ALS", presented at IBIC'20, Santos, Brazil, Sep. 2020, paper FRAO03, unpublished.

[13] Xilinx CLK104 Board, https://www.xilinx.com/support/documents/boards_and_kits/zcu216/ug1437-clk104.pdf

[14] LBNL DSBPM repository, https://github.com/BerkeleyLab/LBNL-DSBPM

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THPDP080

[15] G. Huang, L.R. Doolittle, Y.L. Xu, and J. Yang, "Low Noise Digitizer Design for LCLS-II LLRF", in *Proc. North American Particle Accelerator Conf. (NAPAC'16)*, Chicago, IL, USA, Oct. 2016, paper TUPOA40, pp. 364–366. doi:10.18429/JACoW-NAPAC2016-TUPOA40

[16] Xilinx AXI Stream Interconnect, https://www.xilinx.com/products/intellectual-property/axi4-stream_interconnect.html

[17] LBNL Cell Controller repository, https://github.com/BerkeleyLab/LBNL-Cell-controller

[18] J. M. Weber, J. C. Bell, M. J. Chin, W. E. Norum, and G. J. Portmann, "Advanced Light Source High Speed Digitizer", in *Proc. IBIC'20*, Santos, Brazil, Sep. 2020, pp. 132–135. doi:10.18429/JACoW-IBIC2020-WEPP16

[19] LBNL HSD/BPM repository, https://github.com/BerkeleyLab/LBNL-Digitizers

[20] Micro-Research Finland Oy, http://www.mrf.fi

[21] LBNL Dual EVG repository, https://github.com/BerkeleyLab/LBNL-Dual-EVG

[22] LBNL Event Fanout/Receiver, https://github.com/BerkeleyLab/LBNL-EVFR