Design of the Electron-Ion Collider Common Platform and Applications for RF Controls

K. Mernick[1]*, R. Bachimanchi[2], K. Fahey[1], J. Latshaw[2], S. Mai[1], M. McCooey[1], G. Narayan[1], J. Settle[2], F. Severino[1], A. Singh[1]

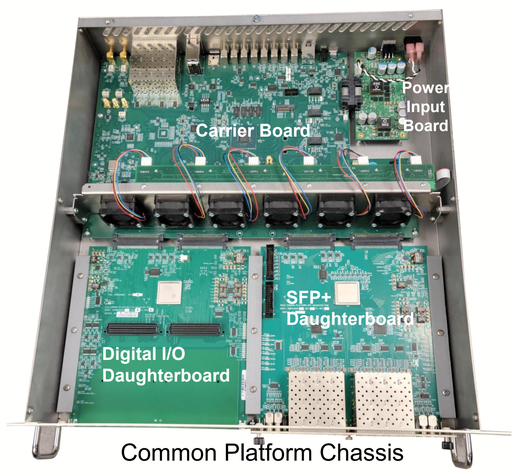
*kmernick@bnl.gov

[1]Brookhaven National Laboratory, Upton, NY, [2]Thomas Jefferson National Accelerator Facility, Newport News, VA

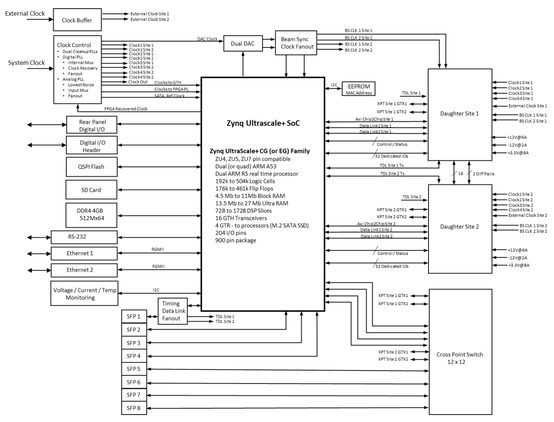
Abstract

The EIC Common Platform is a modular system architecture which will serve as the basis for EIC Accelerator Controls. It consists of an SoC-based carrier board with up to two independent pluggable FPGA-based Daughtercards. Different types of Daughtercards have custom electronics catering to the specific needs of an application. Daughtercards will have FPGA logic to support a common protocol for communication with the carrier board as well as a basic set of features for programming and telemetry. RF Controls applications will use two versions of an RF Digitizer Daughtercard designed by the LLRF team as well as several of the general purpose Daughtercards designed in collaboration by the LLRF, Accelerator Controls, and Instrumentation groups. The system architectures for various LLRF applications using the Common Platform components will be presented.

Chassis, Carrier, and Daughtercard Architecture



Common Platform Chassis



Common Platform Carrier Block Diagram

The EIC Common Platform is a 2U pizza-box, network-attached device chassis with a common carrier board and function-specific daughtercards. This design inherits significantly from the C-AD LLRF Platform[1]. The chassis power input is modular and will be supplied by a rack-level 48 V DC PDU in most cases. Daughtercards include a set of “core” general purpose modules, such as digital IO, baseband multichannel ADCs, and an SFP+ board used for distribution of machine timing and data. Application specific daughtercards include the LLRF digitizers and the BPM processors.

The common carrier board includes a Xilinx Zynq UltraScale+ system-on-chip which interfaces to the accelerator control system, including the Timing Data Link, a high-speed serial link similar to the current LLRF Update Link [2] which provides deterministic timing and low-latency communication between systems. Daughtercards include a common backend based on a Xilinx Artix UltraScale+ which provides the carrier interface and application logic.

The block diagram above highlights key features of the carrier design.

- Xilinx Zynq UltraScale+ FPGA
  - Dual (or quad) ARM Cortex-A53 application processor
  - Dual ARM Cortex-R5 real-time processor
  - Programmable Logic
  - 16 GTH high-speed serial transceivers
    - 8 SFP+ external connections
    - 12x12 crosspoint switch
    - 4 dedicated + 2 crosspoint daughtercard connections
- RAM & non-volatile memory
- Dual Ethernet
- Versatile clocking options
- Debug (serial, JTAG)
- Remote download/configuration at bootup

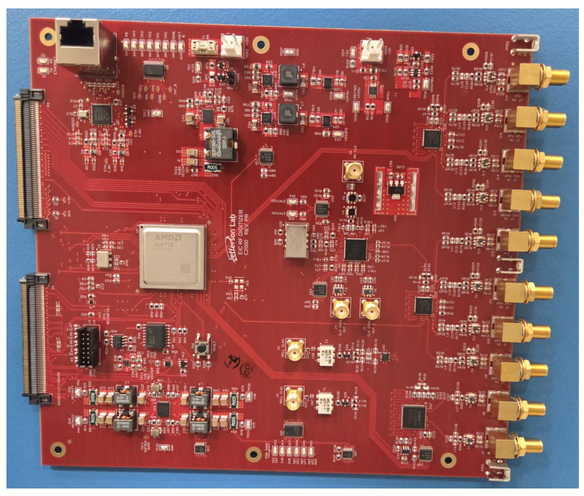
The clocking scheme supports running on an externally supplied system clock for lowest jitter performance or clock recovery from the Timing Data Link. Digital and analog PLLs supply multiple clocks to the carrier and daughter sites. An integrated DDS, using the AD9783 DAC, can generate two beam-synchronous clocks.

RF Digitizer Daughtercard

Two versions of an RF Digitizer daughtercard are planned. Both will use the AD9783 dual channel DAC and AD9653 quad channel ADC, used in the JLab Digitizer-1 and Digitizer-2 designs, as well as LCLS-II and many other LLRF projects. The difference between the two cards is the number of DAC and ADC channels on each. Planned versions are:

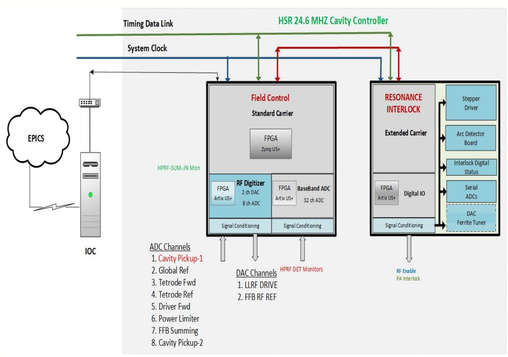
- 8 ADC channels, 2 DAC channels
- 4 ADC channels, 4 DAC channels

The initial prototype of the 8 ADC / 2 DAC version of the digitizer, which was just received recently, is shown at right. Performance characterization is in process, with plans for testing on a cavity in 2026. The firmware design leverages many existing components from the JLab and BNL LLRF systems, as well as common daughtercard components already developed, speeding the system integration process.

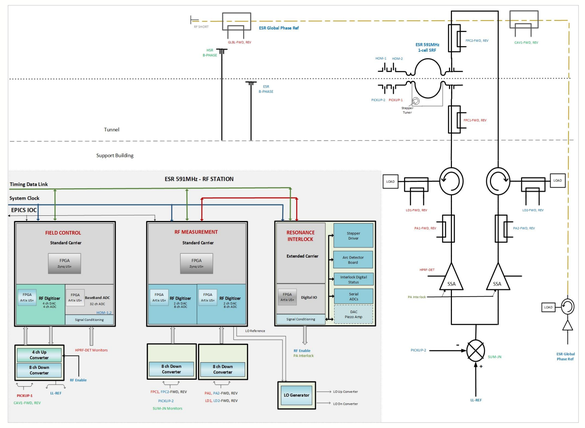


8 ADC / 2 DAC RF Digitizer Daughtercard

Cavity Controller Applications



HSR 24.6 MHz NCRF Cavity Controller

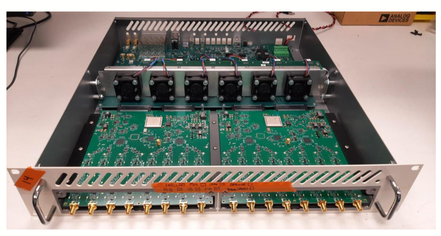


ESR 591MHz 1-cell SRF Cavity Controller

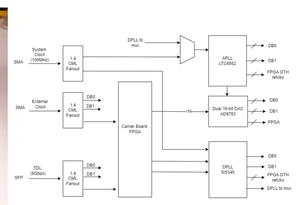
Two examples of the system configuration for cavity controllers are shown in the figures above. The HSR 24.6 MHz cavity configuration on the left is representative of low-frequency NCRF systems (24.6, 49 & 98 MHz). These will use direct sampling on the ADC and DAC. The figure on the right shows the ESR 591 MHz 1-cell SRF cavity configuration. This cavity has two FPCs driven by independent 400 kW solid-state amplifiers. Higher frequency systems will use up and down conversion with ADC IFs around 24 MHz and DAC IFs generally in the 120 to 150 MHz range. These IF frequencies are in a similar range to those used previously at BNL and on other projects (LCLS-II, for example). The 197 MHz systems will use down conversion for the ADC with direct sampling on the DAC.

The resonance and interlock chassis leverages the extended carrier and digital IO daughtercard Common Platform components along with custom breakout and signal conditioning boards to interface to the many devices and IO standards required for this application. The extended carrier provides an interface for an external daughtercard to an existing Common Platform carrier board over a point-to-point fiber link.

Digital IO Daughtercard

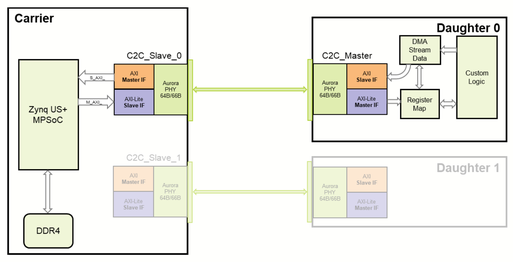


Carrier and 2 BPM Daughtercards



Carrier Clock Sources

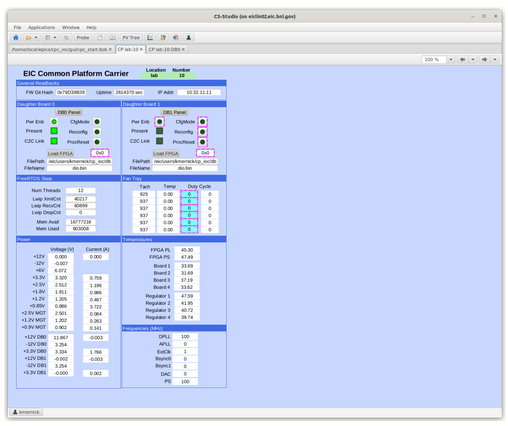
Firmware & Software Interfaces



AXI Chip2Chip Bus Carrier/Daughter Interface

A memory mapped interface between the carrier Zynq processing system and the daughtercard is implemented using the AXI chip2chip bridge over an Aurora serial link. This gives the carrier low-bandwidth AXI4-Lite master access to read/write to the daughtercard register interfaces. The daughtercard implements a high-bandwidth AXI4 master interface for DMA transfer of measurements and other data.

The firmware development process is utilizing the FPGA Firmware Framework (fwk) developed by the DESY MSK group[3]. This standardizes the FPGA development process, promotes modular and reusable designs, and includes git integration. The DesyRDL tool also provides a standardized register definition and interface method and generates output files to simplify the software interface development.



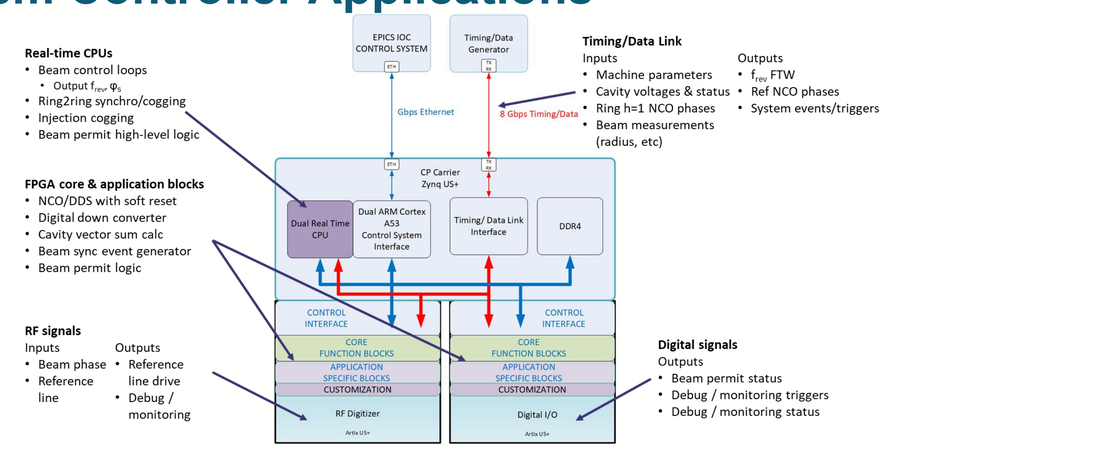
Control System Studio Interface Screen

EIC is transitioning from the locally developed Accelerator Device Object (ADO) based control system, used in RHIC and its injectors, to the widely used, open source, EPICS [4] control system.

The interface from the carrier board to an EPICS IOC is using the Portable Streaming Controller driver (PSCDrv) [5]. This development effort has been greatly advanced by collaboration with colleagues from the NSLS-II who have been using this interface for many years.

An example of an interface screen for carrier telemetry data and basic daughtercard interface control is shown above.

System Controller Applications



HSR System Controller Block Diagram

The system controllers handle necessary machine-level tasks, including implementation of the RF beam control loops for the HSR, synchronization of machines for beam transfers or collision cogging, and other functions. The system architecture for implementing these beam control and synchronization requirements is similar to the approach used in RHIC and its injector complex. An additional function of the system controllers for EIC is to provide compensation for long-term phase drifts. Porting of the RHIC system controller software to run on the Common Platform is in progress.

References

[1] K.S. Smith, T. Hayes, F. Severino, “Concept and Architecture of the RHIC LLRF Upgrade Platform,” PAC 2011

[2] T. Hayes, K.S. Smith, F. Severino, “A Deterministic, Gigabit Serial Timing, Synchronization and Data Link for the RHIC LLRF,” PAC 2011

[3] FPGA Firmware Framework Documentation, https://fpgafw.pages.desy.de/docs-pub/fwk/index.html

[4] The Experimental Physics and Industrial Control System, https://epics-controls.org/

[5] The PSC Driver, https://mdavidsaver.github.io/pscdrv/

This work was supported by the EIC Project and the U.S. Department of Energy, Contract DE-SC0012704 and Contract DE-AC05-06OR23177.

Electron-Ion Collider

Brookhaven National Laboratory

Jefferson Lab

U.S. DEPARTMENT of ENERGY | Office of Science