Proposal for an RCE-based DAQ system for LBNE

M. Convery, M. Graham, G. Haller, R. Herbst, M. Huffer

SLAC National Accelerator Laboratory, Menlo Park, CA 94025

(Dated: February 14, 2013)

ABSTRACT

This document presents a proposal to use the SLAC-developed DAQ toolkit

1 Introduction

The main purpose of the LBNE DAQ system is to read the raw data from the Front End Boards (FEB), which are mounted on the Anode Plane Arrays (APA) inside the cryostat, to build events from the different parts of the detector and to pass these events on to long term storage. The Level 3 requirements for this system include[1]:

- LArFD-L3-DAQ-3: The DAQ shall be capable of receiving raw data from a freely running readout from all detector systems.
- LArFD-L3-DAQ-7: The DAQ shall be designed to collect data continuously
- LArFD-L3-DAQ-8: The DAQ shall perform prompt processing of data

The DAQ-7 requirement is relevant mainly to non-beam physics. As such, it was left out of the requirements at the time of CD-1, which assumed a surface-located Far Detector (FD). Nonetheless, continuous readout remains a valuable goal that would be desirable to have in the final LBNE DAQ system.

The key electonics module that needs to be provided for the back-end DAQ is one that is capable of reading the data streams from each of the APA's and concentrating the data down to a smaller number of high-bandwidth data streams that are then passed to an event-building network. This is commonly needed function in modern HEP experiments that has frequently been addressed with custom modules built explicitly for a single experiment. This may require significant development time. However, the modules produced quickly become obsolete, as available networking technology progresses. This limits the desirability of reusing such modules in subsequent experiments.

The SLAC Research Electronics Group (REG) has developed a solution to this obsolescence problem by producing a set of modules, together with firmware and software, that can be adapted for use in multiple experiments. The development costs are then leveraged over multiple experiments, allowing each of them to benefit from the latest networking hardware, at a significant reduction in development costs. This "DAQ toolkit" uses the modern Advanced Telecommunications Architecture (ATCA)for its physical structure. The key element of the system is the Reconfigurable Cluster Element (RCE), which is based on a Virtex 5 "System on a Chip". A single board combining several of these RCE's can handle very high bandwidths measured in the 100's of Gigabits/second. This system has been adopted in several HEP experiments already, and will likely be adopted by more in the future. The REG is continuing to develop and support new generations of the toolkit to take advantage of new networking equipment as it becomes available.

We are proposing to make use of this toolkit in the DAQ systems to be produced for the LBNE 35 ton prototype and the full Far Detector. The bandwidth available in the current generation of the toolkit far exceeds that of the baseline system based on the Nova Data Concentrator Module (DCM). The increased flexibility afforded by this extra bandwidth may be highly valuable to ensuring LBNE success. Furthermore, leveraging the work already done by the REG as well as benefitting from their support in the future, will provide many benefits to LBNE and may reduce the development costs.

2 The Data Acquisition Toolkit

2.1 Advanced Tele-Communication Architecture and the ATCA Shelf

The ATCA shelf is known historically as the chassis and is by analogy, equivalent to a VME crate. Shelves house the Front-Boards and RTMs described below (see Sections 2.1.2 and 2.1.3). They contain, from front to rear, pairs of slots with each pair housing a Front-Board in the front and the Front-Board's corresponding RTM in the rear. The shelf allows for hot-swap of any board in any slot. Depending on form factor the number of its slot pairs varies from two (2) to sixteen(16). The orientation of those slots also varies, as shelves are offered with either horizontal or vertical orientation. In turn, that orientation affects the flow of air; from either left to right (horizontal), or top to bottom (vertical). Broadly, the shelf is composed of a sub-rack, backplane, filters and cooling devices (fans). The subrack provides the infrastructure to contain the Front-Boards and RTMs described below. This includes guide rails, ESD discharge, alignment, keying, and backplane interface. Backplanes are passive circuit boards which carry the connections between slots. Although somewhat more complicated in detail, for this document, those connections can be partitioned into three logical groups: power, control and differential data pairs. The topology for both power and control connections is invariant of backplane. However, in order to accommodate different applications the connection topology of data pairs can vary. Two commonly used topologies are the dual star and full mesh.The backplane (and ATCA) is protocol agnostic with respect to the usage of these differential pairs with the choice delegated to the shelf's specific Front-Boards.

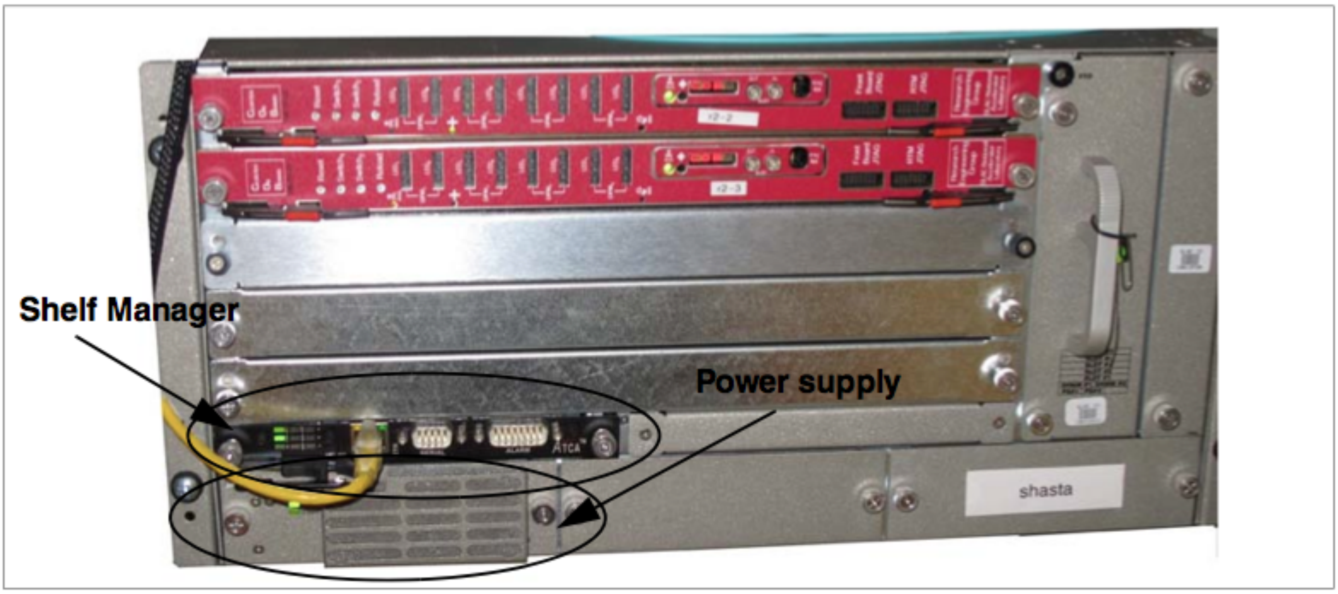


FIG. 1: Front view of 5-slot ATCA shelf.

This photograph is of a COTS1 shelf purchased from ASIS [7]. It has a horizontal orientation within its corresponding rack with airflow from left to right. It contains a replicated, full mesh backplane. Two of its five front slots are populated with Front-Boards, while its unused slots are populated with dummy air baffles. Note the RJ45 connector located on the front-panel of its Shelf-Manager (ShMC). This provides the shelf manager access to the Ethernet from which control and monitoring (through IPMI) of the shelf would be accomplished. Further, note the integral power supplies. These supplies are not required by the ATCA standard, but are provided by ASIS as a convenient feature for bench-top usage. The same shelf viewed from the rear is illustrated in Figure 2.

2.1.1 The Front-board

The Front-Board constitutes the heart of the ATCA eco-system. From a shelf's perspective that board is simply a PCB board, 8U wide x 280 mm deep and which plugs into one of its front slots. That board, although following ATCA mechanical and electrical interface

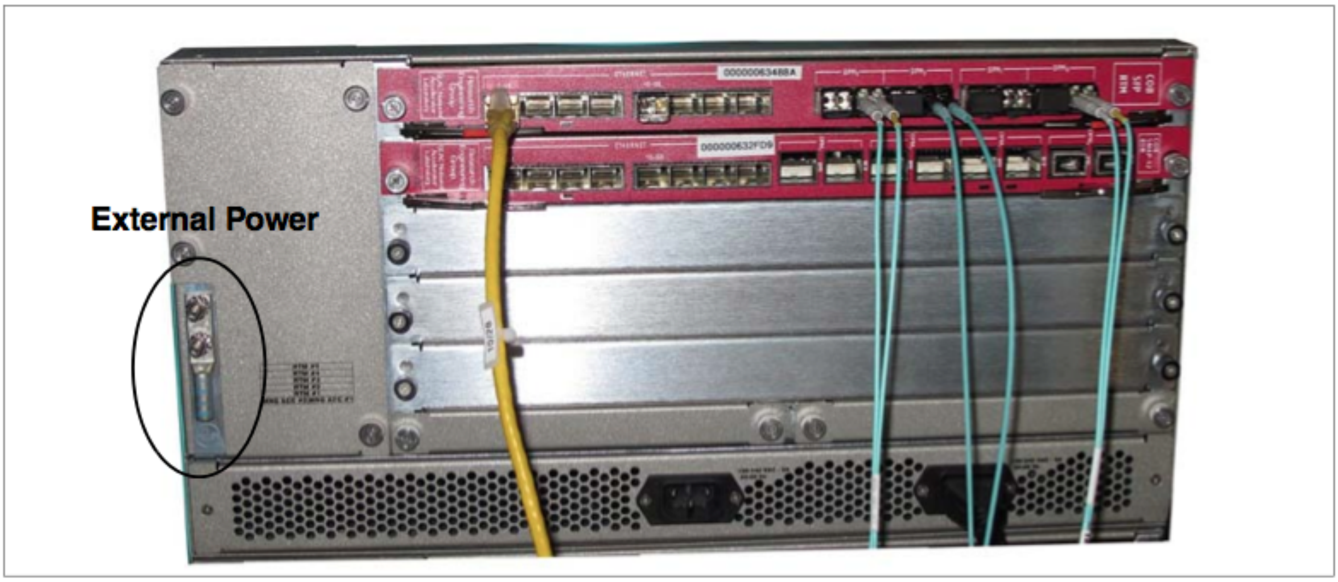


FIG. 2: Back view of 5-slot ATCA shelf.

standards, contains logic which is application specific. The board's rear side contains three logical Zones. Zones 1 and 2 connect directly to a shelf's backplane. Zone 1 provides access to shelf power (+48 VDC) as well as the I2C communication channels which the board uses to communicate with its shelf manager. Zone 2 provides access to the high-speed, differential pairs connecting boards together. The area encompassed by Zone 3 is application defined, but reserved for connections to the board's RTM. LBNE requires one application specific Front-Board. That board is described in Section 2.2. A photograph of a representative Front-Board, showing connectivity to an RTM (using PICMG 3.8) is illustrated in Figure 3.

2.1.2 Rear Transition Module

The RTM (Rear-Transition-Module) is simply a PCB board, 8U wide x 70 mm deep which is used to extend a Front-Board and house a board's external, I/O interface. The RTM shares the same hot-swap model as the Front-Board and specifies an identical pitch (1.2"). This allows the RTM to reuse the same panel, handle switches, and LEDs as its Front-board. The RTM connects to its Front-board through Zone 3. The form of that connection is application specific. However, if power for the RTM is necessary, it must be provided by the Front-board and must be brought through Zone 3. The ATCA specification is somewhat ambiguous with respect to the maximum power drawn by an RTM. Zone 3 is

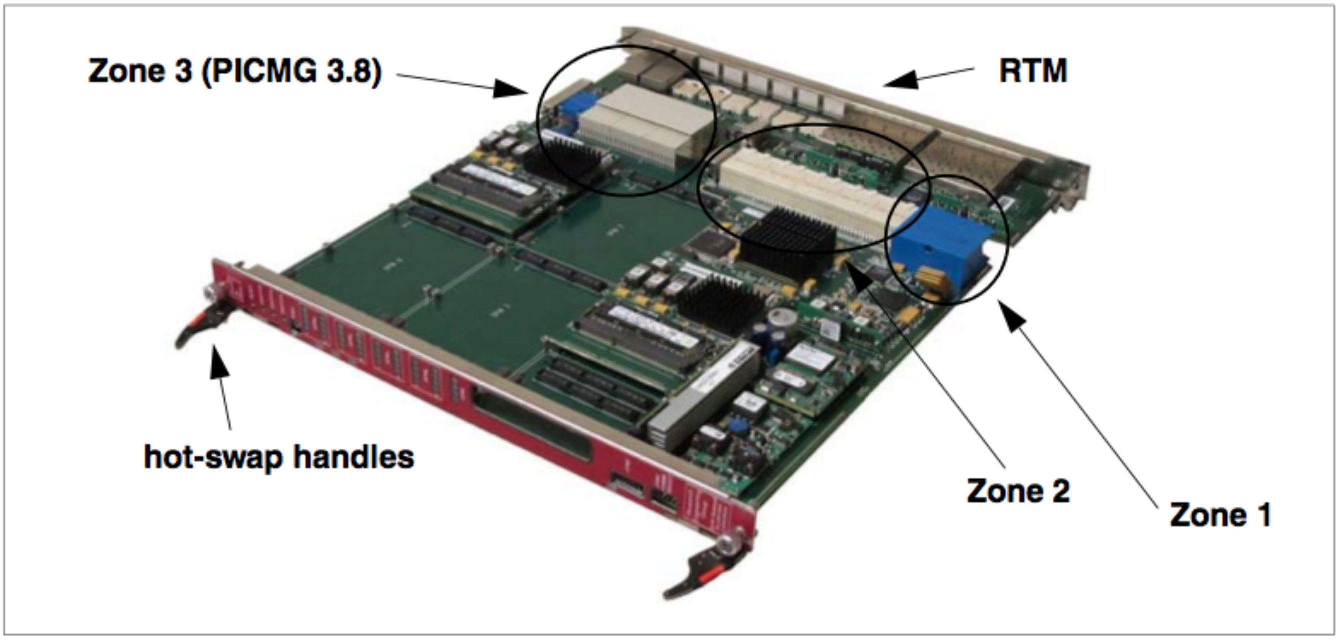


FIG. 3: Representative ATCA front-board.

populated with two connectors, one for power and one for signal. Power provided through the power connector is +12 VDC and that connector also contains pins for JTAG as well as I2C support. The I2C channel is expected to be used by the Front-Board for control of the RTM's hot-swap switch as well as its front panel LEDs. The signal connector provides up to 120 differential pairs. How those pairs are assigned between Front-Board and RTM is considered application specific. However, for LBNE's Front-Board, each one of its four DPM bays is assigned 1/4 of those pins or thirty (30) pairs (see Section 2.2).

2.2 Cluster-on-Board

The COB (Cluster-On-Board) is an 8U, ATCA compliant Front-Board with a PICMG 3.8 Zone 3. Functionally, the COB serves as a carrier board for the RCEs hosting the firmware and software developed for LBNE (see Section 2.4). Those RCEs are mounted on mezzanine boards (see Section 2.5.2), which in turn plug into Bays on the COB. Bays are connected to the COB's two separate, independent Interconnects as well as its Zone 3 connectors. Interconnects provide arbitrary, high speed communication paths between the elements contained on the bay's mezzanine boards, both (it is important to note), inter and intra COB. Although rated up to 300 watts, when fully populated with five mezzanine boards, a COB draws closer to 120 watts. This board is one deliverable from SLAC's R & D program on high-speed DAQ. As such, LBNE simply purchases this board and from its perspective, that board consequently requires neither design nor development. A photograph of that COB (in preproduction form) with its five bays occupied is shown in Figure 4.

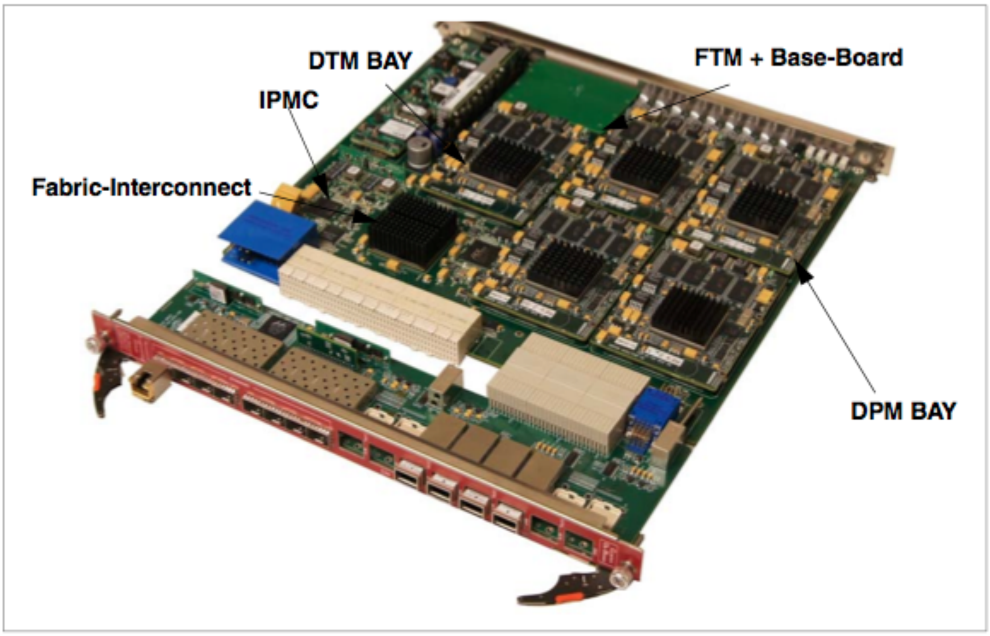


FIG. 4: Pre-production COB.

The COB contains five (5) bays; one (1) DTM bay and four (4) DPM bays. Although all bays share identical form factors and connectors, they can be differentiated, primarily by how they connect to Zone 3, with the DTM connecting only to its power connector and the DPM only to its signal connectors. In turn, those connections determine the function of their corresponding mezzanine boards. The DTM, interacting with its shelf manager, manages the health and safety of both COB and RTM, while DPMs acquire and process data originating from the RTM. Those data, their interface, acquisition and processing are all intended to be application specific.

The mezzanine board plugged into the DTM (Data-Transport-Module) bay contains one RCE as well as the COB's IPM Controller (IPMC). The IPMC is the element responsible for monitoring the underlying health and safety of the COB as well as its corresponding RTM. It is also responsible, in conjunction with its corresponding shelf manager, for board and RTM activation/deactivation. It performs all these activities by interacting with various components on the COB, specifically with the RCEs contained within the COB's five bays.

That interaction is accomplished through dedicated, local I2C busses. The IPMC is a SOC (System-On-Chip), containing a dedicated ARM based (M3) processor. That processor runs de-facto, industry standard Pigeon-Point IPMC firmware and software [41], suitably modified to control and monitor the specific functionality of the COB.

Although in capability and form no different than any other RCE, the DTM's RCE has the fixed, dedicated responsibility for managing both of the board's interconnects. For this purpose it contains specific firmware and software. For example, as one responsibility, it must maintain the configuration and supervise the 10G-Ethernet switch contained within the fabric interconnect. That switch's management interface is a single lane PCIe. To communicate with this switch, the RCE contains a PCIe Protocol-Plug-In (firmware, see Section 2.4) as well as the tools (software) to configure and monitor that switch. Note, however, that while the DCM's RCE has predefined, base responsibilities it also remains accessible for user applications.

The mezzanine board plugged into a DPM (Data-Processing-Module) bay contains two (2) RCEs. Each DPM provides connections to thirty (30) differential pairs originating from the RTM, but carried through the COB's Zone 3 signal connector. The mapping of those thirty pairs to the mezzanine board's two RCEs is arbitrary and determined by application. The function of either RCE is determined not only by the mapping of those thirty pairs, but by the firmware and software it contains.

2.3 Interconnects

The fabric interconnect contains, as its principal feature, a local, 10-Gigabit Ethernet (10-GE). Packets are switched on that network using a commercial ASIC. That ASIC is a fully compliant Layer-2, 10G-Ethernet switch. Although fully provisioned for buffered transfer, switch operation is, by default, cut-through with an ingress/egress latency of less than 200 Nanoseconds. It is also a fully managed switch with a PCIe interface connected to the DTM's RCE. Through its interconnect the COB's RCEs appear as nodes on that Ethernet. The interconnect allows its physical network to be extended to both nodes and networks external to the COB. Those networks could be, for example, other COBs residing in the same shelf, or even nodes physically disjoint from both COB and its shelf.

The base interconnect's principal function is to manage and distribute synchronous timing to the COB's five bays. Note that unlike the fabric interconnect the protocol distributed over this interconnect is application specific. In further contrast to the fabric interconnect which functions identically independent of the shelf slot it occupies, the base interconnect has slot dependent responsibilities. This is a consequence of the fact that while the fabric interconnect uses ATCA's fabric interface, the base interconnect uses its base interface. That interface employs a backplane topology that is fixed by the standard at dual-star. ATCA refers to slots at its roots as Hub slots and slots at its leaves as Node slots. Necessarily, the behavior of a board, specifically its base interconnect, must vary depending on whether it occupies either a hub or a node slot. While boards in node slots need only distribute timing locally, boards occupying node slots must distribute timing not only locally, but also to other boards occupying its shelf. In short, while occupying a hub slot the base interconnect drives its base interface, but while occupying a node slot receives timing.

2.4 Reconfigurable Cluster Element

The RCE (Reconfigurable-Cluster-Element) is a bundled set of hardware, firmware and software components. Together, those components form a generic computational element targeted to process efficiently, with low latency, those kinds of data found passing through HEP DAQ systems. Those data have in common three features which make specific, somewhat, competing demands on the functionality of any such element. Those features are:

- Highly parallel: Data which are massively parallel are most naturally also processed in parallel, requiring computational elements which scale in cost, footprint and power. Those elements, in order to manage the flow of their data both efficiently and coherently, communicate together. This necessitates a communication mesh which shares the same scaling properties as the elements themselves.
- Inhomogeneous: As those data typically originate with their corresponding detector they are carried necessarily over a variety of media employing various inhomogeneous protocols. The element's I/O structure, must support, naturally, without sacrifice of performance that diversity.
- Transient: Transient data arrive at an element once, to be either transformed or reduced before immediately exiting the element. Such data are not typically amenable

to caching strategies and require elements whose optimal computational model emphasizes a performanent efficient I/O structure, coupled strongly to a large, low latency memory system over raw processor speed.

The RCE is optimized for those three features. Physically, one element can be contained in a footprint of less than 32 cm^2, typically draws less than eight (8) watts, costs (in small quantities) around $750 and contains a native 10-Gigabit Ethernet interface. Elements are connected through a commercial, commodity ASIC containing a 64 channel, Layer-2, cut-through1, Ethernet switch [38]. The combination of elements and switch define a Cluster and the nature of ethernet as well as functionality within that switch allows for the composition of arbitrary numbers of cluster hierarchies. For example, from the RCE perspective, the COB represents a single cluster of nine (9) RCEs and its ATCA shelf is simply a container for a single level hierarchy of up to fourteen (14) nine node clusters. A block diagram of the major physical features of the RCE is illustrated in Figure 5.

The principal implementation feature of the RCE is in its reuse of System-On-Chip (SOC) technology, specifically, member's of Xilinx Virtex-5 FX family [55]1. As such, the RCE is neither processor, FPGA or DSP. Instead, it can be simultaneously any combination of the three. Within its fabric the FPGA contains both soft (user defined) and hardened (manufacture defined) silicon. That fabric is configured automatically on POR (Power-On-Reset) and is either downloaded directly from images previously stored on the FPGA's configuration (platform) flash, or indirectly through the RCE's JTAG interface. Note also that the platform flash is itself programmed through the RCE's JTAG interface. The RCE employs standard Xilinx tools and software to program the FPGA. Xilinx refers generically to its set of different, hardened silicon as resources. Among the more important of those resources are high speed serializers/deserializers, I/O adapters, DSP tiles, dual-port RAM and of course, its processor. The RCE allocates the processor as well as a modest number of additional resources and soft silicon for its CE (Cluster-Element). The CE has exclusive use of, but interfaces indirectly with its external DDR3 memory and micro-SD flash system. Memory is packaged as SO-DIMM and the micro-SD flash is removable, allowing its capacity to be determined by user application.

The BSIs (Boot-Strap-Interface) principal function is to reset the CE. However, it also contains the initial configuration information necessary for the CE's bootstrap loader to boot its processor. The BSI is outside the CE so that its configuration may be retained

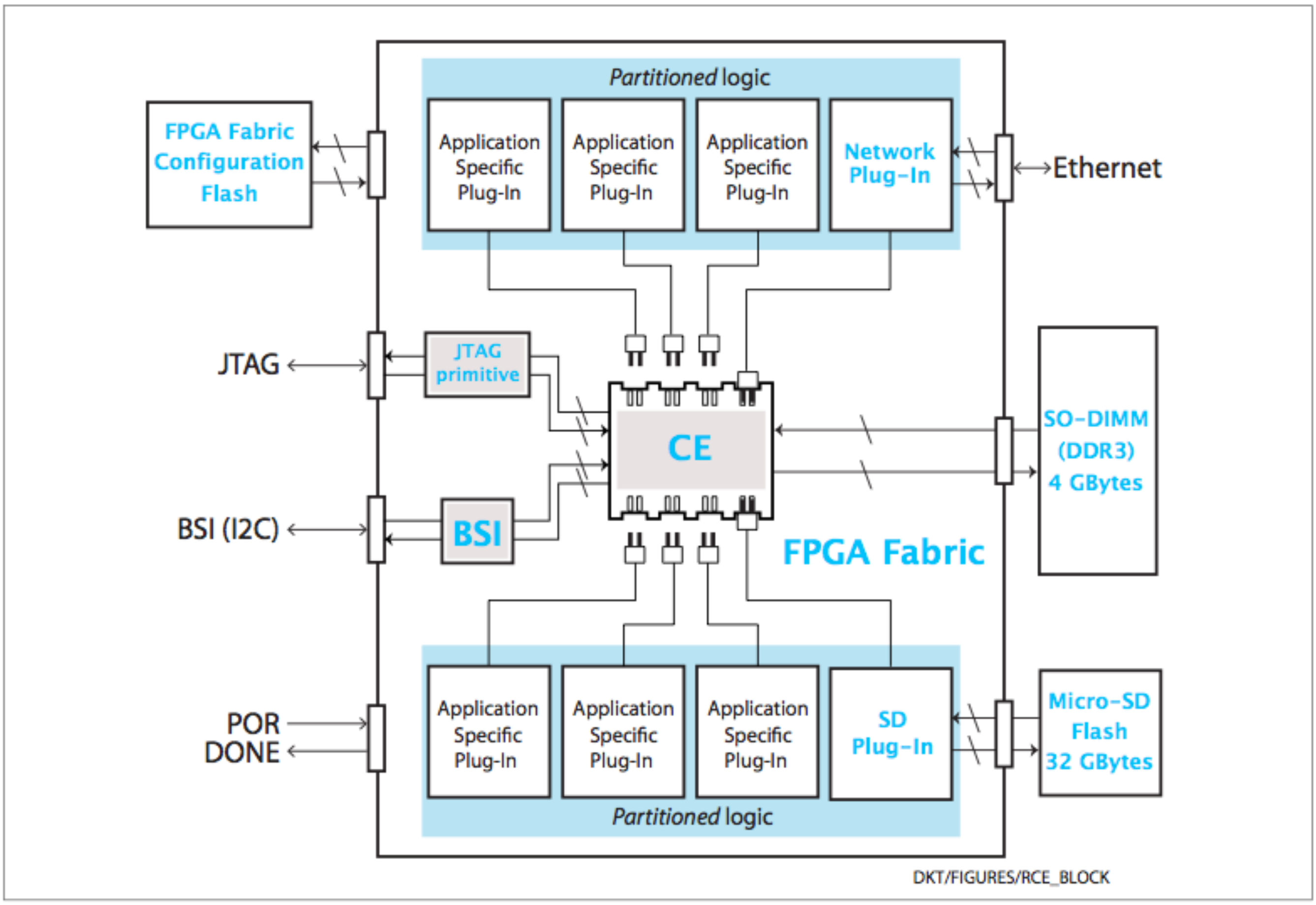

Partitioned logic
FPGA Fabric Configuration Flash
Application Specific Plug-In
Application Specific Plug-In
Application Specific Plug-In
Network Plug-In
Ethernet
JTAG
JTAG primitive
CE
SO-DIMM (DDR3) 4 GBytes
BSI (I2C)
BSI
FPGA Fabric
POR
DONE
Application Specific Plug-In
Application Specific Plug-In
Application Specific Plug-In
SD Plug-In
Micro-SD Flash 32 GBytes
Partitioned logic
DKT/FIGURES/RCE_BLOCK

FIG. 5: Block diagram of the RCE.

over resets of the CE. External to the FPGA the BSI appears as a standard I2C device and receives its command and control through that interface. Note, for the COB, that device is controlled and monitored through its IPMC.

To provide isolation between system and user firmware and insure reproducible behavior, system firmware is partitioned [53] away from application specific logic. System firmware is defined as the CE, the BSI, JTAG support and both Network and SD Plug-Ins. The CE, which is both at the heart of the entire RCE and contains a significant fraction of the user's intellectual investment is described in Section 2.5. The remainder of the fabric, both hardened and soft silicon is reserved for application specific logic. That logic and its relationship with the CE is described below.

Although both user defined and implemented, any application specific logic, does of course require information exchange between it and its CE1. The interface model which allows such exchanges is the plug and socket. To follow that model, the user wraps their implementation specific logic with a thin veneer of system provided firmware. That wrapper is the plug and the combination of user logic and its plug is called a Protocol-Plug-In or PPI. When wrapped, that logic is now capable of being plugged into any of the eight predefined sockets on the CE. And once plugged in, both PPI and CE are now able to exchange information.

Although bundled with its base system the RCE itself takes advantage of this model to glue its Ethernet and SD interfaces to the CE. Both are good examples of one class of PPIs which must interface outside their FPGA. Such PPIs when plugged into their CE have as their closest analogy the classic I/O device and processor model. However, unlike that model the PPI model coupled with the resources offered by the FPGA fabric provides an essentially unlimited way to either customize or mold the CE to arbitrary devices and protocols. Of course, the user is not limited to using the fabric and its resources solely for I/O. One can define PPI whose sole purpose is to take advantage of the DSP tiles and combinatoric logic of the FPGA to process rather than transfer data. LBNE can use this functionality to its advantage in performing its feature extraction.

2.5 The Cluster Element

The essential function of the CE is as a platform which serves as an application specific nexus for the data both received and transmitted through the RCE's application specific PPIs. As such, the CE can be considered as both a hardware1 and software platform. As a hardware platform its principal blocks are illustrated in Figure 6. As a software platform its corresponding services are described in Section 2.5.1.

Its principal implementation blocks are its Memory Controller, Crossbar and Processor: The Memory Controller: Interfaces the RCE's external memory with the CE's Crossbar. It is a soft controller, derived from an existing Xilinx DDR2 design, but tailored for usage of low latency, DDR3 memory. The controller allows addressing of up to four (4) Gbytes of memory. It is clocked at 320 MHZ, has separate, internal, 64-bit, read and write datapaths providing roughly 5 Gbytes/second of either read or write bandwidth.

- The Crossbar: The Crossbar interconnects memory controller, processor1, and up to eight (8) PPI sockets allowing for autonomous, concurrent transfers between all three types of entities and providing arbitration for when those transfers might collide. The

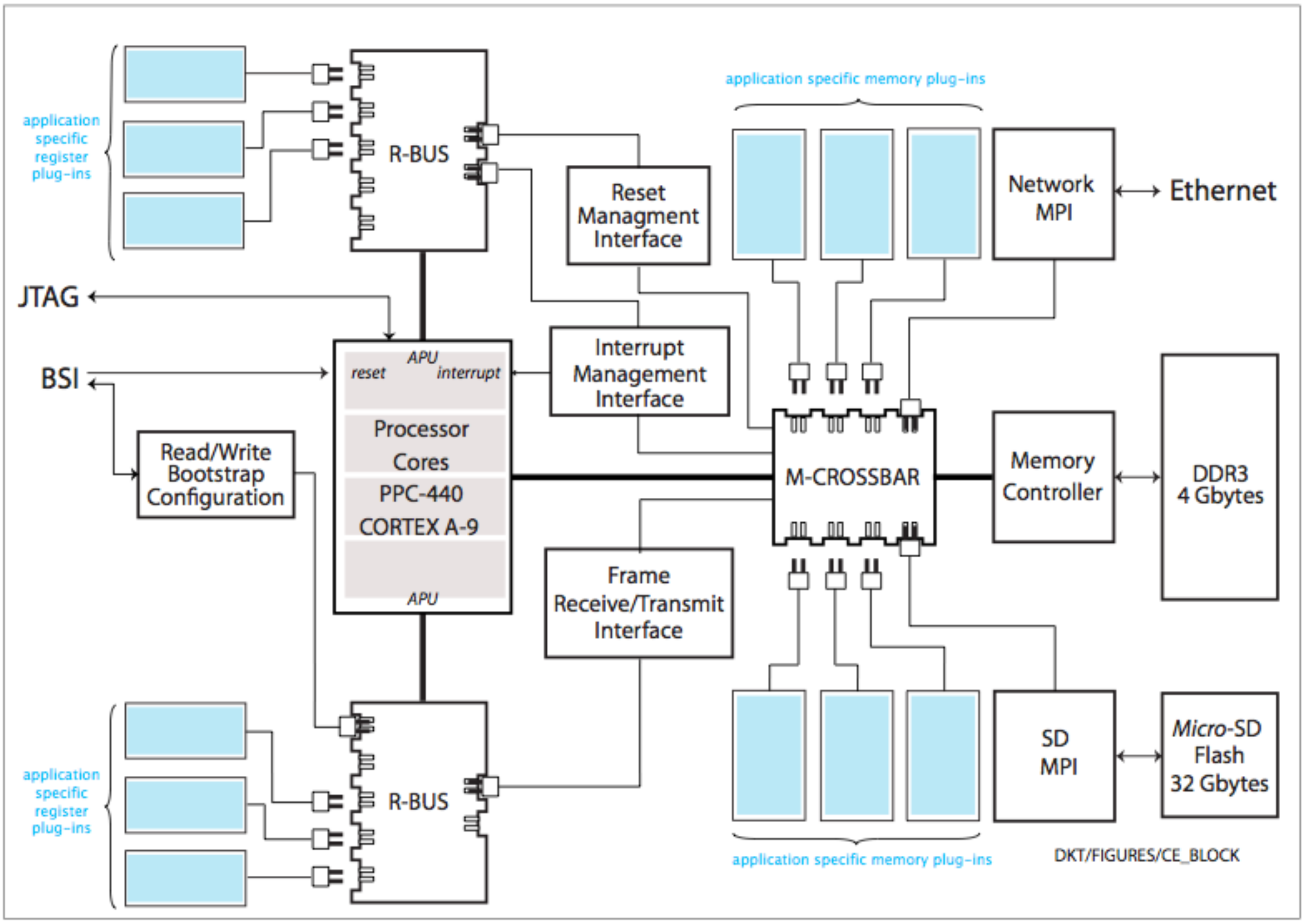

application specific register plug-ins
R-BUS
Reset Managment Interface
application specific memory plug-ins
Network MPI
Ethernet
JTAG
BSI
APU
reset
interrupt
Processor Cores
PPC-440
CORTEX A-9
APU
Interrupt Management Interface
Read/Write Bootstrap Configuration
M-CROSSBAR
Memory Controller
DDR3 4 Gbytes
Frame Receive/Transmit Interface
application specific register plug-ins
R-BUS
SD MPI
Micro-SD Flash 32 Gbytes
application specific memory plug-ins
DKT/FIGURES/CE_BLOCK

FIG. 6: Block diagram of the CE.

crossbar is clocked at the same rate as its memory controller (320 MHZ) and contains internal, separate, 128-bit, read and write datapaths. Its core is hardened silicon [56], but suitable enhanced with purpose built firmware which glues the eight PPI sockets to that core.

- The Processor: A 32-bit, PowerPC-440, superscaler, single core, RISC processor with separate 32 Kbyte data and instruction caches [56]. It is clocked at 475 MHZ. In addition to the three busses connected to the crossbar, the processor contains another, separate, independent, 128-bit wide bus called its APU bus [56]. One side is connected to the processor and its other side is an interface to the FPGAs fabric. This bus is unique in that it interacts directly with the processor's registers and data cache, bypassing its memory completely. Essentially, it allows the user to extend the processor's instruction set with application specific logic implemented in its fabric.

Taking advantage of this feature, the CE uses the APU to control and manage its PPI sockets through a set of instructions which transfer data into and out of a socket directly from either registers or cache. This provides a very effective, low latency, performanent mechanism to transfer small amounts of data between processor and PPI. A similar mechanism is used for large data transfers, where data, rather than passed to and from the socket by value, are now passed to and from by reference. The socket autonomously takes care of transferring the data pointed to by that reference either to or from the PPI. Arbitrary transactions which interleave data by both value and reference are supported.

2.5.1 Software Services

The RCE includes bundled software to accelerate and leverage the development of application specific code for the CE. Some set of this software is linked to and executes with those applications (system resident software), while a subset is in the form of tools that operate cross-platform. Any and all system resident software is distributed with each RCE and if used, is dynamically linked to its corresponding applications. Remote tools and any software updates have a well defined release and distribution mechanism. JIRA is used for a bug-tracking and reporting system. Here is a summary of the software services bundled with the RCE:

- Bootstrapping: A generic bootstrap loader which allows, on reset, transfer to arbitrary code based on an externally controlled configuration parameter called its current vector (contained within the BSI). The code loaded and executed by the loader is assumed stored in the RCE's micro-SD device. The code pointed to by any specific vector is called a bootstrap. Bootstraps may be either standalone code or Version/Issue: 1.1/1 code which loads and transfers control to other code (a secondary loader). The CE may contain and transfer control to an arbitrary number of different bootstraps. For LBNE, on reset, control is transferred to a secondary bootstrap which starts up RTEMS (see below).

- Operating/System: Although the CE is itself O/S agnostic, its system resident software is not and depends on functionality best provided by the services of an underlying

O/S. In order to not compromise the RCE's innate performance a Real/Time (R/T) kernel offered the best compromise in satisfying that functionality. That kernel is RTEMS. RTEMS has a fully provisioned set of multi-tasking services as well as being both compact and efficient. It also maintains POSIX compliant interfaces, easing the burden of porting third-party software. However, perhaps most importantly, it is an Open-Source product with no licensing issues. RTEMS is described in additional detail in [31].

- Persistency: Access to micro-SD based media using its bundled PPI. That media is formatted as FAT-16 and is used by the CE for storage of system code and configuration (see bootstrapping above). However, that media is available directly to applications for storage of their own application specific code and configuration.

- Networking: Includes a complete TCP/IP stack. The stack's MAC layer is satisfied by the RCE's bundled 10G-Ethernet PPI. The user interfaces to that stack are POSIX compliant.

- Linking: The same dynamic linker used to bridge system and user code.

- PPI support: Interrupt and reset support for an application's PPI.

- Debugging: Support for both local and remote debugging. Local debugging (SMD) interfaces to JTAG through standard Xilinx tools. Remote, network based, debugging uses the GNU interface.

- Diagnostics: Built-in self-tests as well as diagnostics. These are included on the CE as an alternate boot image providing the ability to rescue or repair inadvertent burns of the micro-SD media. Development employs the GNU cross-development environment [34].

2.5.2 Mezzanine Board

The mezzanine board is one physical implementation of the abstract RCE described above in Section 2.4. It is a PCB board (100 mm x 80 mm) which hosts either one or two elements of RCE. A mezzanine board plugs into any one of the five bays contained on a COB.

Power (+6 VDC) to this board is applied using two separate, but identical connectors. One connector is assigned to each element of the board. Those connectors provide, in addition to power, a presence sense pin as well as an enable pin for that power. The board's two, internal PDS (Power-Distribution-Systems) takes that input voltage, divides it down and distributes the necessary, well regulated voltages to each element. Each PDS can source 25 Watts.

A high-speed, high density, differential connector carries signals between the COB and the elements of the mezzanine board. Those signals include:

- To and from RTM (thirty pairs).
- To and from the Fabric interconnect (sixteen pairs).
- To and from the Base interconnects (eight pairs).
- JTAG.
- To and from the IPMC (I2C); one per element.

On each of its two I2C channels the board contains, in addition to the element's BSI various I2C devices which provide the following information:

- PDS status.
- Board and die temperatures.
- Element serial number (64 bit).
- Persistent, configuration information (MAC addresses, element wiring, etc.)

The COB's IPMC uses that information to plug and play with its bays, including their activation as well as in the monitoring of their health and safety. To illustrate both mezzanine concept and its relationship to the RCE, a photograph of the prototype (single element) GEN-II RCE, mounted in a mezzanine board is shown in Figure 7.

2.5.3 Pretty Good Protocol

The Pretty Good Protocol (PGP) is a VHDL module which facilitates the bi-directional transmission of frame based messages over a two-wire physical link.

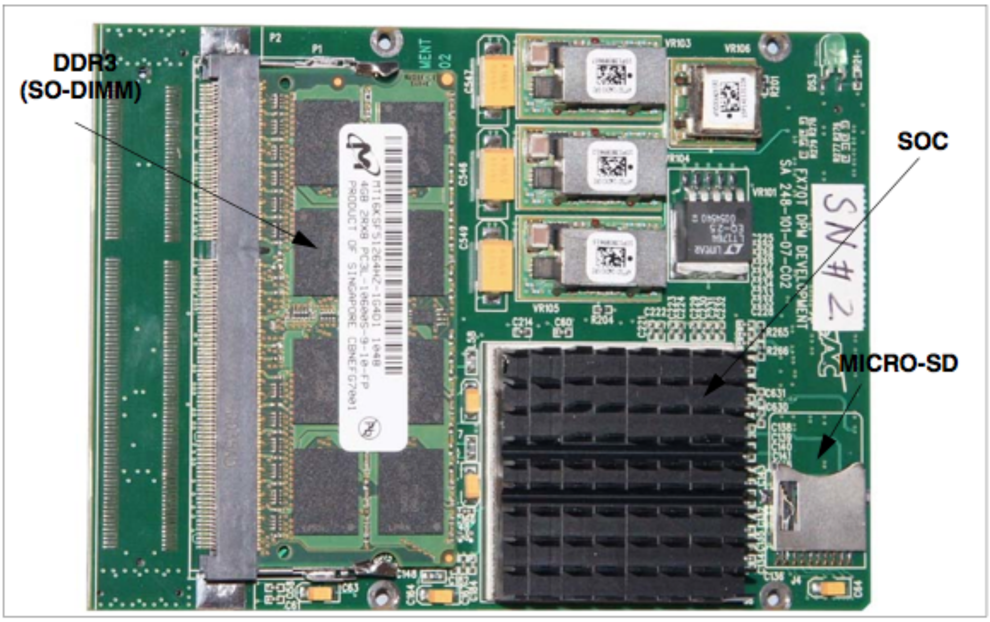


FIG. 7: A single element RCE on a mezzanine board.

3 Implementation of RCE-based DAQ for LBNE

The elements of the DAQ-toolkit described in the previous section can be easily applied to the LAr TPC for LBNE. The block diagram of a possible configuration is shown in Fig. 8. We define the ”front-end DAQ” as everything between the (cold) FPGA and the ATCA shelf; from the ATCA-shelf onward is referred to as the ”back-end DAQ”. The primary goal of this document is to propose a solution for the back-end DAQ and so, for this purpose, we will assume that the signals come into the back-end DAQ from the output of the front-end board (FEB) FPGA, each of which collects the output of $8 \times 16 = 128$ TPC wires.

The basic structure of the back-end RCE-based DAQ is fairly straightforward. The data from the ADCs is encoded (possibly using the PGP protocol; see Appendix **??**) in the FEB FPGA and driven out of the cryostat to a ”flange board” which converts the electrical signal to an optical signal. The flange board is an optional step, but one which allows the back-end DAQ crates to be conveniently placed without worrying about signal degradation. The optical signal is then sent to the RTM which interfaces with the COB. RTM designs with up to 48-channel fiber optic inputs exist and are currently in use by LCLS and for LSST development (???this is made up???). The RTM also incorporates the output to the DAQ PC farm via 8 x 10 Gbps ethernet. The RCEs on the COB can be used to perform event building or even some level of pattern recognition.

One of the DPMs on each COB will function as the trigger and timing interface. The external timing and (for the 35t) trigger signals will be received via optical fiber to the RTM and will be distributed to the other DPMs and out (through the RTM) to all of the FEBs. A scheme exists for distributing the timing and trigger over the PGP link.

Below, we discuss some specific issues with the implementation in the 35t prototype and full LBNE, as well as some ideas for the front-end DAQ.

3.1 Flange Board

The electrical signals will be brought out of the cryostat and converted to optical signals just outside the flange on custom built flange boards. Each flange board houses optical drivers to handle the electrical-optical conversion and to transmit the optical signals to the back-end DAQ. The copper side of the flange board will depend on the choices made by the cold electronics group; the fiber side use SNAP-12 connectors. The size and number of flange boards depends on the number of connections needed and mechanical specifications. A block diagram of a two connector board is shown in Fig. 9. Table I assumes a single pair of receiver-transmitter pairs per flange board.

	35t	Full LBNE
Total Channels	∼2.3k	∼307k
Number of APAs	4 (?)	120
Number of FEBs	18	2400
Transition Boards	2	200(????)
RTM+COB Boards	1 + 1	50 + ?
ATCA Crates	1	4 (14-slot)

TABLE I: DAQ-related quantities for the 35t and full LBNE (as of Jan. 2013 design). The assumption is that we will readout one wire/FEB.

3.2 DAQ Layout for 35t Prototype

The 35t prototype TPC will have $\sim 100\times$ fewer channels than the full LBNE TPC and additionally will be externally triggered to observe cosmic rays. The trigger rate is estimated

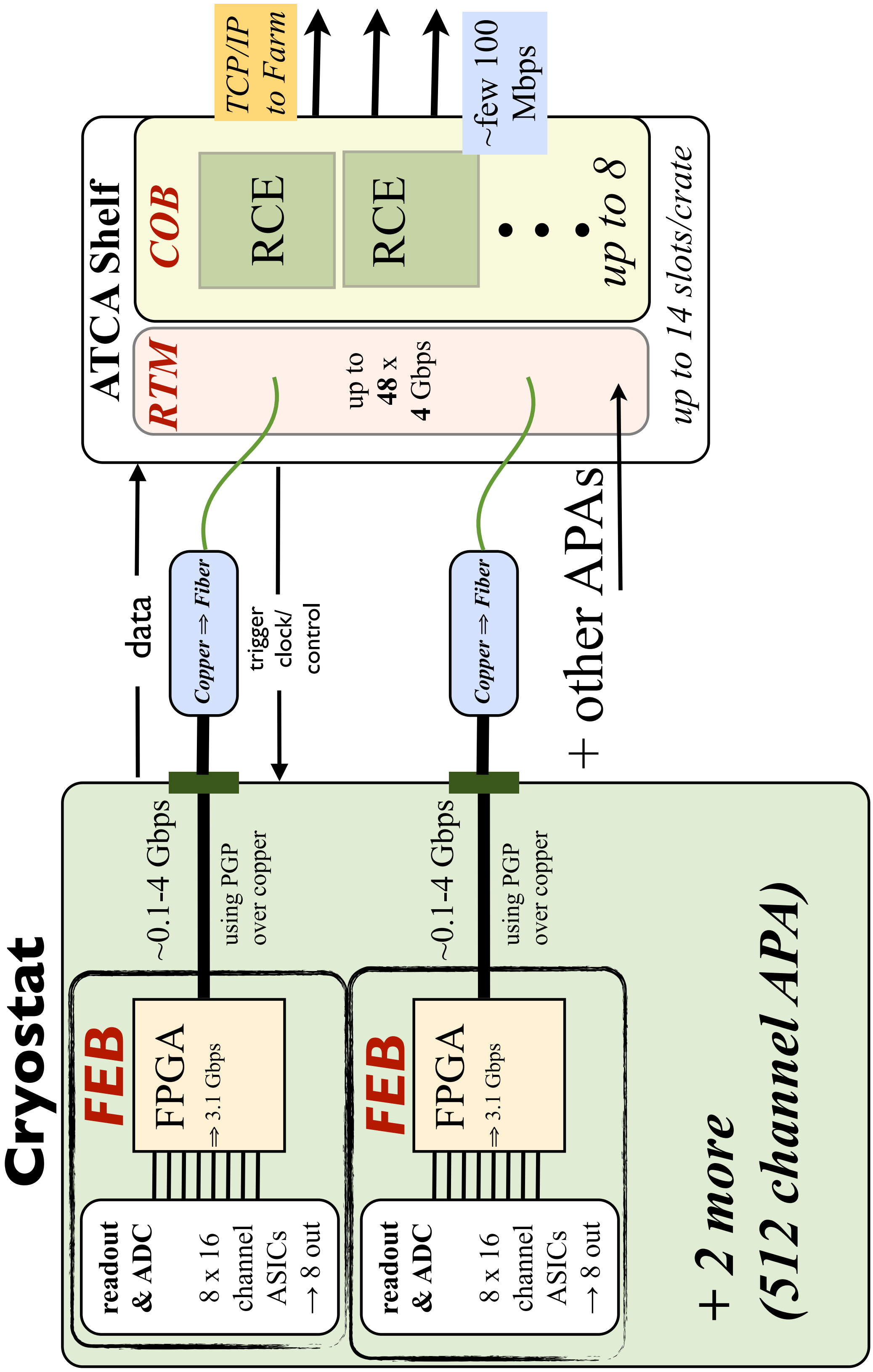


FIG. 8: Block diagram of the RCE-based DAQ for a single TPC APA.

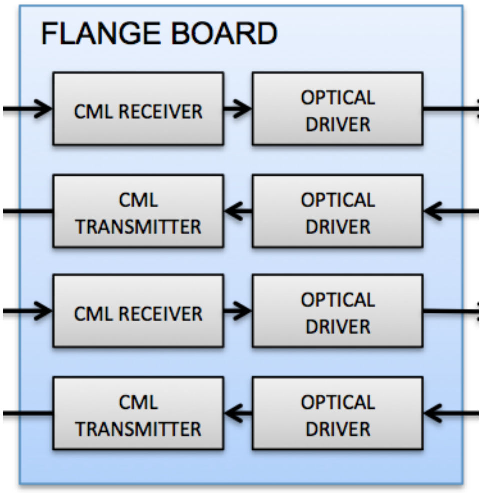


FIG. 9: Block diagram of a flange board with two pairs of transceivers.

to be $<$ 1kHz. If we run out a single wire/FEB from the cryostat, then the entire TPC can be read into a single COB. Bringing more wires/FEB would require more COBs; the maximum number of input channels/RTM is 48. For 8 wires/FEB, 4 COBs are likely required, since the trigger and timing signals will take up one input to the RTM each.

From the DAQ perspective, the photon system will consist of 32 digitized signals from the output of a CAEN digitizer (some of these may be sync or trigger signals). We will likely want to convert these signals to PGP/fiber at an early stage to best utilize existing firmware. The signals from the photon system will be read into a separate COB.

In total, even if multiple wires/FEB are read out, the 35t DAQ will easily fit into a single ATCA shelf.

	35t	Full LBNE
Total Channels	XXXX	XXXX

TABLE II: Data rates etc ... (as of Jan. 2013 design).

3.3 Full LBNE

3.4 Comparision of RCE-based vs DCM-based Backend DAQ Systems

3.5 High-speed Data Links From Cold FPGA to Backend DAQ

...possibilities and our plans on this ...

4 Schedule and Budget

The proposed schedule and estimate of costs for the 35t prototype RCE-based DAQ is shown in Fig. 10.

This proposal uses the Gen-3 COBs which are expected to be available in August (???) at the latest. The Gen-3 COBs will use the Virtex ZYNQ SoC (series-7) which has many advantages over the series-5 previously used. Probably the most important feature for our purposes is that it is distributed with with a Linux distribution. This makes software development much simpler; we can develop our algorithms in C++ on a Linux PC and simply copy them directly to the RCEs. While there is overhead involved with running Linux (and it cannot come near the speed of running algorithms in firmware), we expect that it will be fast enough for the few hundred Hz trigger rates expected in the 35t prototype.

The budget includes 4 DAQ test stands composed of a 2-slot ATCA crate (with power), a COB and RTM, a simplified flange board, and an emulation board. The emulation board will consist of a Virtex-5 FPGA programmed to output data with format and rate similar to that expected from the TPC FEB and transmit it to the flange board via copper. The test stands will allow us to develop the DAQ system at places other than at SLAC. One of these test stands can likely be used for the 35t DAQ system. Much of the development needed to get the first test-stand working will be covered under a SLAC LDRD so, for project purposes, is considered to be available at no cost.

There are a number of tasks necessary for the complete working system that can be done semi-independently of the full ATCA-based DAQ. Preliminary work on some of these items can start immediately on test boards or PCs which is advantageous because we don't expect to receive the Gen-3 boards until August. All of the tasks listed can be performed by non-SLAC universities or labs with a low-level of SLAC support.

- Timing/Trigger: The external clock and trigger must be integrated into the RTM/COB and distributed to the FEBs. Much of this work likely requires a COB. Details on the clock and trigger signals may be necessary for RTM design.

- Event Building: If desired, some level of event building can be performed in the COBs before sending to the PC farm. This work can be done by a physicist and can be started at any time with C++ code on a Linux PC and be transfered to the RCEs when ready.

- PGP Implementation on FEB: Once the FEB is laid out, we can used a PCIe-based PGP card to program the cold FPGA (up to timing/trigger, which will require the COB). This task requires an EE.

- Photon System: The output and timing/trigger needs will need more fleshing out. This may require either a different RTM or flange board than the TPC. The design of this can be done independently of the COB.

	Hours	Cost ($1k$)
Physicist	XXXX	0
Engineer	XXXX	XXXX
Technician	XXXX	XXXX

TABLE III: Total number of work hours and assumed cost.

Materials	35t Prototype	Full LBNE
Physicist	XXXX	0
Engineer	XXXX	XXXX
Technician	XXXX	XXXX

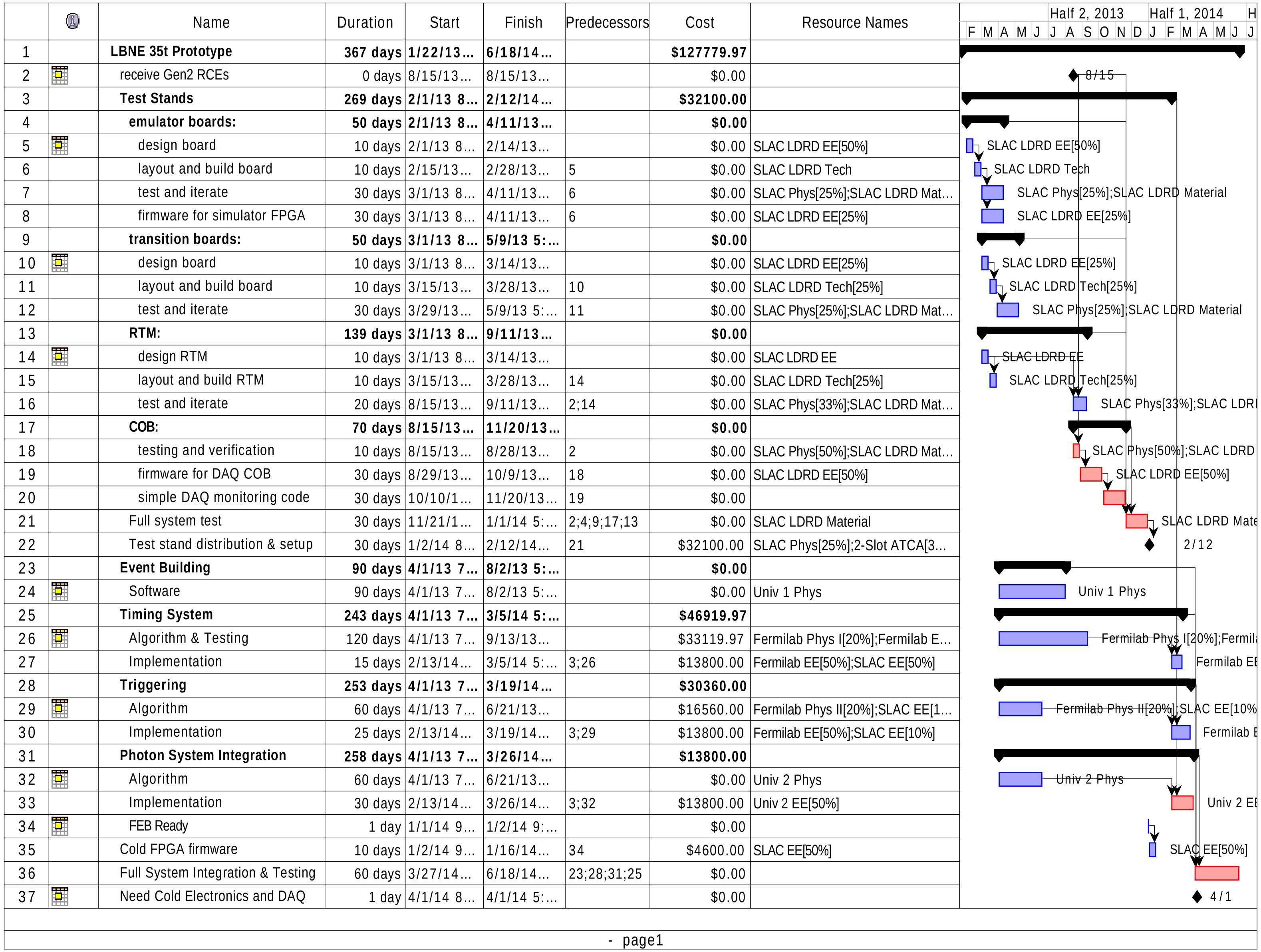
TABLE IV: Estimated materials cost of the back-end DAQ for 35t and the full 10kt LBNE.

5 Conclusions

... why there is no choice be to go with us ...

6 References

[1] B.Baller *et al.*, LBNE Document 3747-v5,”LAr-FD Level 2 Programmatic and Scientific Requirements and LAr-FD Level 3 Requirements”



	ⓘ	Name	Duration	Start	Finish	Predecessors	Cost	Resource Names
1		**LBNE 35t Prototype**	**367 days**	**1/22/13...**	**6/18/14...**		**$127779.97**	
2	▦	receive Gen2 RCEs	0 days	8/15/13...	8/15/13...		$0.00	
3		**Test Stands**	**269 days**	**2/1/13 8...**	**2/12/14...**		**$32100.00**	
4		**emulator boards:**	**50 days**	**2/1/13 8...**	**4/11/13...**		**$0.00**	
5	▦	design board	10 days	2/1/13 8...	2/14/13...		$0.00	SLAC LDRD EE[50%]
6		layout and build board	10 days	2/15/13...	2/28/13...	5	$0.00	SLAC LDRD Tech
7		test and iterate	30 days	3/1/13 8...	4/11/13...	6	$0.00	SLAC Phys[25%];SLAC LDRD Mat...
8		firmware for simulator FPGA	30 days	3/1/13 8...	4/11/13...	6	$0.00	SLAC LDRD EE[25%]
9		**transition boards:**	**50 days**	**3/1/13 8...**	**5/9/13 5:...**		**$0.00**	
10	▦	design board	10 days	3/1/13 8...	3/14/13...		$0.00	SLAC LDRD EE[25%]
11		layout and build board	10 days	3/15/13...	3/28/13...	10	$0.00	SLAC LDRD Tech[25%]
12		test and iterate	30 days	3/29/13...	5/9/13 5:...	11	$0.00	SLAC Phys[25%];SLAC LDRD Mat...
13		**RTM:**	**139 days**	**3/1/13 8...**	**9/11/13...**		**$0.00**	
14	▦	design RTM	10 days	3/1/13 8...	3/14/13...		$0.00	SLAC LDRD EE
15		layout and build RTM	10 days	3/15/13...	3/28/13...	14	$0.00	SLAC LDRD Tech[25%]
16		test and iterate	20 days	8/15/13...	9/11/13...	2;14	$0.00	SLAC Phys[33%];SLAC LDRD Mat...
17		**COB:**	**70 days**	**8/15/13...**	**11/20/13...**		**$0.00**	
18		testing and verification	10 days	8/15/13...	8/28/13...	2	$0.00	SLAC Phys[50%];SLAC LDRD Mat...
19		firmware for DAQ COB	30 days	8/29/13...	10/9/13...	18	$0.00	SLAC LDRD EE[50%]
20		simple DAQ monitoring code	30 days	10/10/1...	11/20/13...	19	$0.00	
21		Full system test	30 days	11/21/1...	1/1/14 5:...	2;4;9;17;13	$0.00	SLAC LDRD Material
22		Test stand distribution & setup	30 days	1/2/14 8...	2/12/14...	21	$32100.00	SLAC Phys[25%];2-Slot ATCA[3...
23		**Event Building**	**90 days**	**4/1/13 7...**	**8/2/13 5:...**		**$0.00**	
24	▦	Software	90 days	4/1/13 7...	8/2/13 5:...		$0.00	Univ 1 Phys
25		**Timing System**	**243 days**	**4/1/13 7...**	**3/5/14 5:...**		**$46919.97**	
26	▦	Algorithm & Testing	120 days	4/1/13 7...	9/13/13...		$33119.97	Fermilab Phys I[20%];Fermilab E...
27		Implementation	15 days	2/13/14...	3/5/14 5:...	3;26	$13800.00	Fermilab EE[50%];SLAC EE[50%]
28		**Triggering**	**253 days**	**4/1/13 7...**	**3/19/14...**		**$30360.00**	
29	▦	Algorithm	60 days	4/1/13 7...	6/21/13...		$16560.00	Fermilab Phys II[20%];SLAC EE[1...
30		Implementation	25 days	2/13/14...	3/19/14...	3;29	$13800.00	Fermilab EE[50%];SLAC EE[10%]
31		**Photon System Integration**	**258 days**	**4/1/13 7...**	**3/26/14...**		**$13800.00**	
32	▦	Algorithm	60 days	4/1/13 7...	6/21/13...		$0.00	Univ 2 Phys
33		Implementation	30 days	2/13/14...	3/26/14...	3;32	$13800.00	Univ 2 EE[50%]
34	▦	FEB Ready	1 day	1/1/14 9...	1/2/14 9:...		$0.00	
35		Cold FPGA firmware	10 days	1/2/14 9...	1/16/14...	34	$4600.00	SLAC EE[50%]
36		Full System Integration & Testing	60 days	3/27/14...	6/18/14...	23;28;31;25	$0.00	
37	▦	Need Cold Electronics and DAQ	1 day	4/1/14 8...	4/1/14 5:...		$0.00	

- page1

FIG. 10: Schedule and budget for the 35t RCE-based DAQ.