ProtoDUNE RCE-based TPC readout system

DUNE-doc-XXX October 27, 2016 v1.0

1. Introduction

The readout of the ProtoDUNE TPC wires, prior to being received by PCs in the backend DAQ, consists of the electronics directly attached the APAs, inside the cryostat (the *cold electronics*) and the electronics outside the cryostat, either directly on the cryostat flange or in a rack (the *warm electronics*). This proposal addresses the warm part of the TPC readout and how it receives data from the cold electronics (the **F**ront-**E**nd **B**oards), manipulates the data, and delivers it to the backend DAQ.

This proposal follows closely the design of the 35-ton prototype, where all digitized data from the TPC was sent to a set of System On Chips where it was checked for errors, time-stamped, aggregated, formatted, compressed, buffered, and then sent on the backend DAQ over TPC/IP upon receiving an external trigger. There are a number of differences in the proposed design and the 35-ton design, however, and they will be pointed out in the following sections.

2. Far detector TPC readout requirements

The main functional requirements for the TPC warm readout electronics including the readout processing elements and the on-flange electronics (WIB) are given in the list below. This list focuses mostly on the DAQ-related requirements, but includes the non-DAQ requirements for the on-flange electronics.

- Data Flow:
  - receive, perform error checking, and buffering of the TPC ADC data stream from the front-end electronics (via the protocol specified for the COLDATA chip, see docdb-415)
  - receive trigger signals from an external source ; the DAQ system must be able to handle beam-on trigger rates at least 25Hz
  - send the data in a programmable-width time window surrounding the trigger time to the backend (via Ethernet)
  - perform active reduction in the bandwidth of the data (e.g. compression, zero suppression). The ability to try a variety of these techniques is highly desirable
- Timing Distribution:
  - receive the 50MHz system clock, the 2MHz digitization commands and other timing signals as specified by docdb-415 and distribute them to each FE
- Front-End Configuration & Monitoring:
  - send the required configuration data to the FE
  - receive read-back from the FE to verify configuration and monitor status
- Non-DAQ (warm flange board specific):
  - power distribution

- JTAG
- external calibration

3. Warm Interface Board

From an electronics point-of-view, the protoDUNE flange will consist of 5-slot ‘crate’, where the connectors on the warm side of the feed through form the ‘backplane’ into which the boards plug when inserted into the ‘crate’ assembly. These connectors serve to *send* the power, timing, and configuration down to the FEBs and *receive* the 1.25-Gbps (line rate) signals from the FEBs. The boards in the flange crate (the Warm-Interface-Boards or WIBs) will further multiplex the signals into 5-Gbps streams (or higher) and send the data to the TPC warm readout portion of the DAQ. Details on the WIB-DAQ interface can be found in DUNE doc-db-1394.

Channels/FEB	128	Total/protoDUNE = 15360
Data bandwidth/FEB	3.07 Gbps (12 bits * 2 MHz)	Packed ADC data; doesn’t include header Total/protoDUNE = 368.4 Gbps
Number of links/FEB	4 @ 1.25 Gbps	Total/protoDUNE=480
FEBs/APA	20	Total/protoDUNE = 120
APAs/protoDUNE	6	
Beam duty cycle	4.8s on/ 19.2s off	
Particle rate beam-on	100 Hz	maximum
channels/RCE	256 (2 FEBs)	
Bandwidth-in/RCE	8 Gbps	
RCEs/COB	8	Channels/COB = 2048
COBs/protoDUNE	8	One is only partially filled

Table I: Some basic parameters for the protoDUNE TPC readout

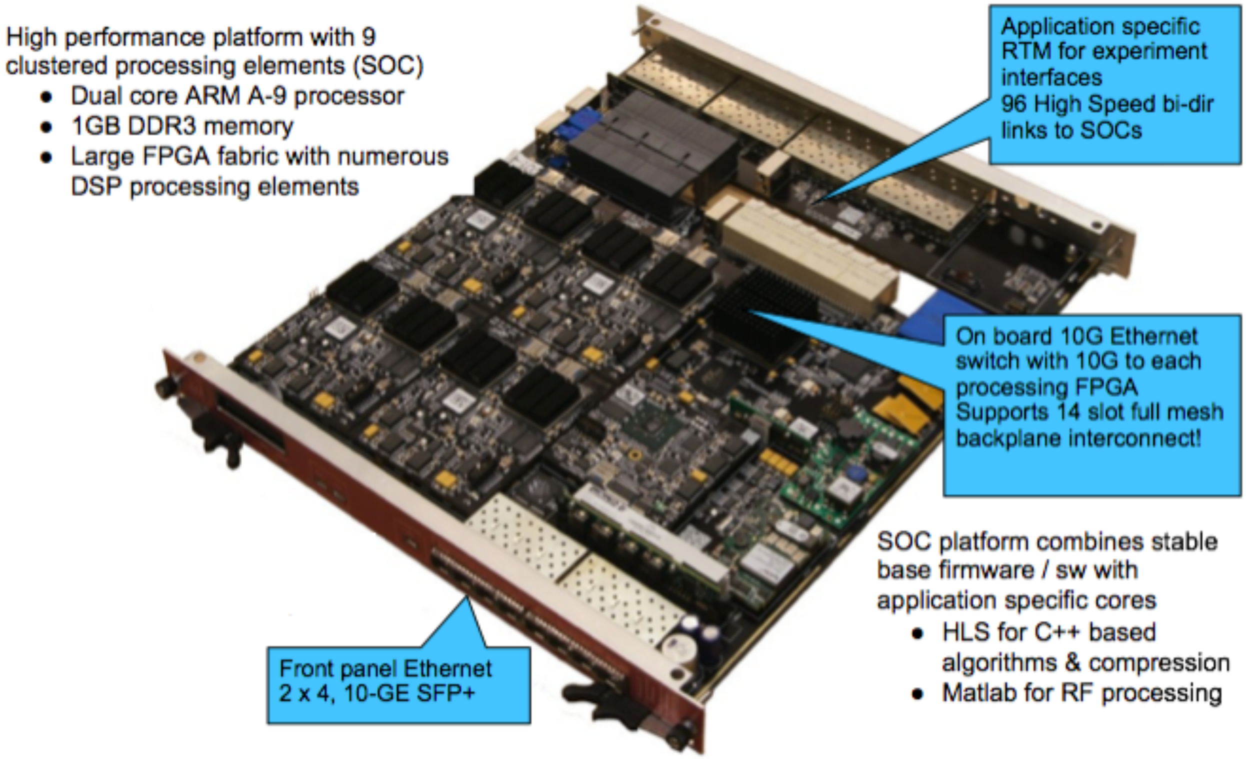
High performance platform with 9 clustered processing elements (SOC)
• Dual core ARM A-9 processor
• 1GB DDR3 memory
• Large FPGA fabric with numerous DSP processing elements
Application specific RTM for experiment interfaces
96 High Speed bi-dir links to SOCs
On board 10G Ethernet switch with 10G to each processing FPGA
Supports 14 slot full mesh backplane interconnect!
SOC platform combines stable base firmware / sw with application specific cores
• HLS for C++ based algorithms & compression
• Matlab for RF processing
Front panel Ethernet
2 x 4, 10-GE SFP+

Figure 1: A COB with RTM, 4 DPMs and DTM.

4. RCE-based readout

The data from the WIBs are received by LArTPC data processors called RCEs (Reconfigurable Computing Elements), which are housed, in industry-standard aTCA crates on COB (cluster-on-board, see Figure 1) motherboards that are designed at SLAC for a wide range of applications. The RCEs are part of a network of field programmable gate arrays (FPGAs) that buffer the full raw data, zero-suppress it for passing to the trigger and accept requests for data-fetching from the trigger. The FPGAs in the RCEs are from the Xilinx Zynq family and contain a full Linux processor system on the chip. They facilitate the high-speed data transfer from firmware into

DRAM memory that is accessible from Linux. A fast data-transfer network using the Ethernet protocol is used on the COBs and in the aTCA crates to allow for development of more sophisticated zero-suppression algorithms for improved supernova acquisition.

The interface with the front-end is provided via the aTCA compliant rear-board, which is dubbed the RTM (Rear Transition Module). This is an application-specific board that is custom made depending on the physical characteristics of the connections to-and-from the FE (e.g. fiber or electrical connections, number and types of transceivers, timing & trigger connections, etc).

For the 35ton LArTPC prototype, we designed a system that connected the front-end boards to the RCEs bidirectionally via a copper-to-optical conversion board on the flange. In that design we used a 1:1 FEB/RCE ratio, so that each RCE handled the data from and clock and configuration to a single FEB (128 channels). The RCEs took in an external trigger that was applied to select what incoming data was sent to the backend.

The setup will be similar for protoDUNE, but for the following changes:

- each RCE will receive data from at least 2 FEBs (256 channels)
- each RCE will compress the data in firmware as it is received
- the direct configuration of the FEBs will be handled (probably) by the FPGA on the flange board;

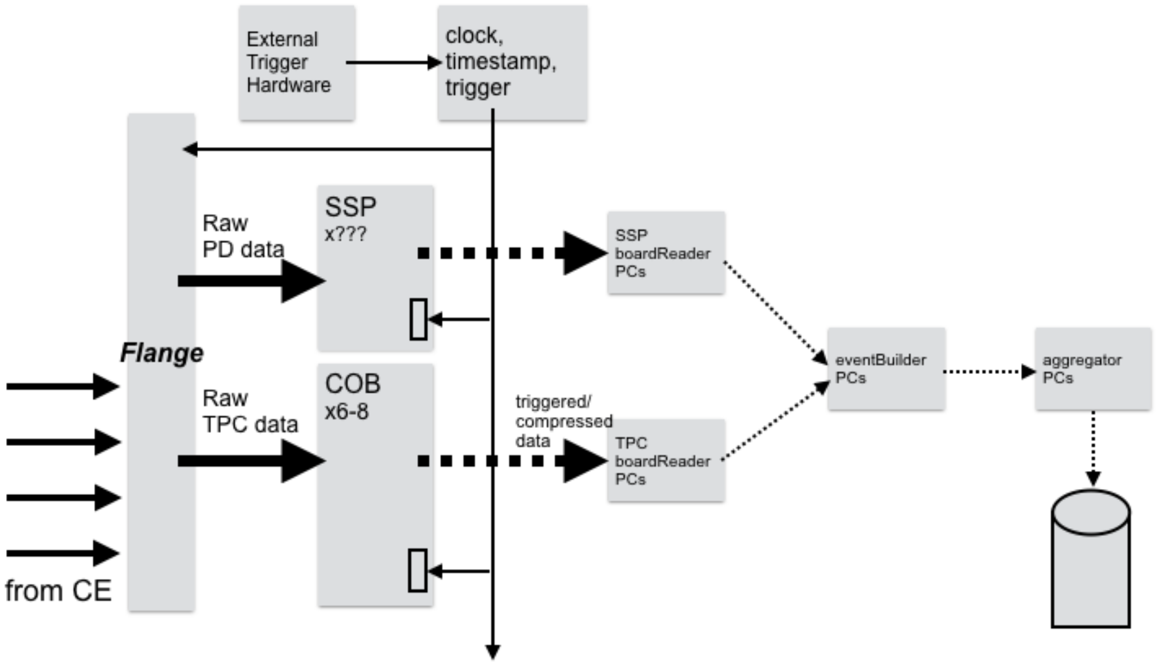

Figure 2: High-level sketch of the protoDUNE DAQ system design

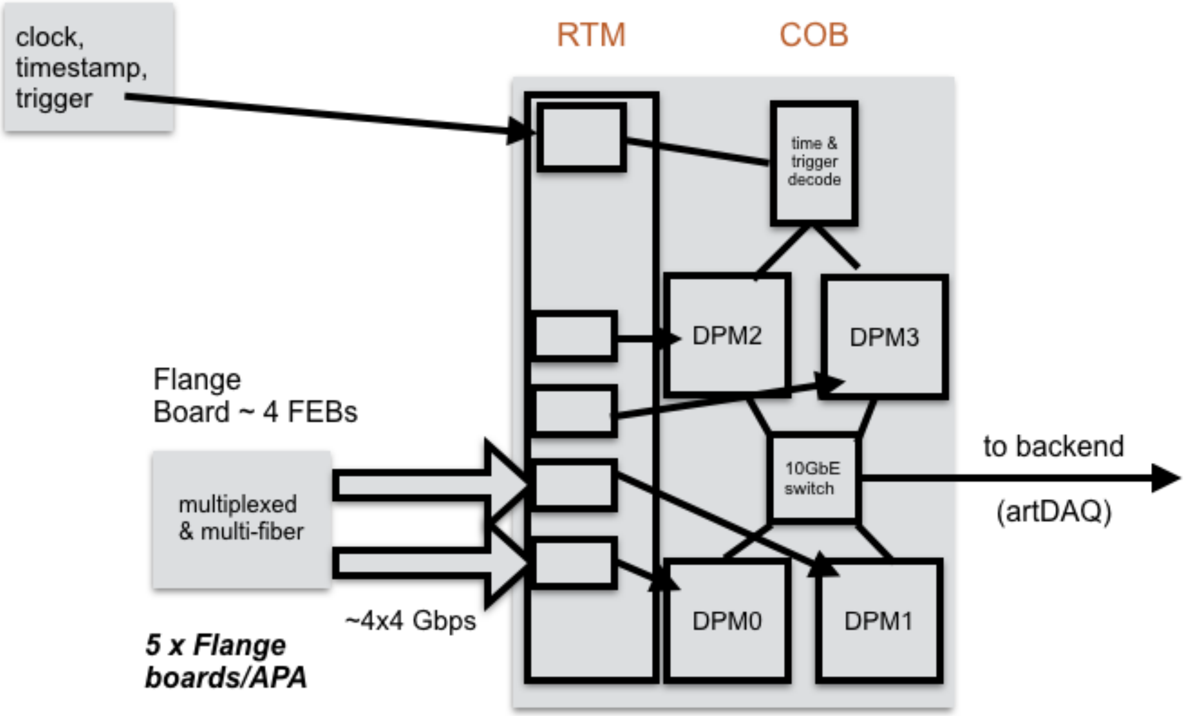

Figure 3: More detailed sketch of the warm TPC readout for protoDUNE.

- the system clock will be generated and distributed independently of the RCE system; the RCEs will still be a consumer of the clock and will still put a global (GPS) timestamp on data as it is received from the FE.

A sketch of the entire system design (FE to backend) is shown in Figure 2, with a somewhat more detailed sketch of the warm TPC readout shown in Figure 3.

4.1 Data flow through the RCEs

As mentioned above, each RCE receives data from two of the WIBs 5-Gbps links, each link carrying the data from one FEB. The data content of each link, including headers and error bits, is roughly 3.5 Gbps. The 12-bit ADC data on each link is sent (serialized) at 2 MHz, 128 channels at a time. The job for the RCE is to:

- deserialize the data
- block it up into larger chunks (we chose 1024 ticks/chunk)
- compress the chunk of data
- DMA the data into the DRAM
- buffer until a trigger comes (or throw away after a timeout)
- send the triggered data to the backend over TCP/IP

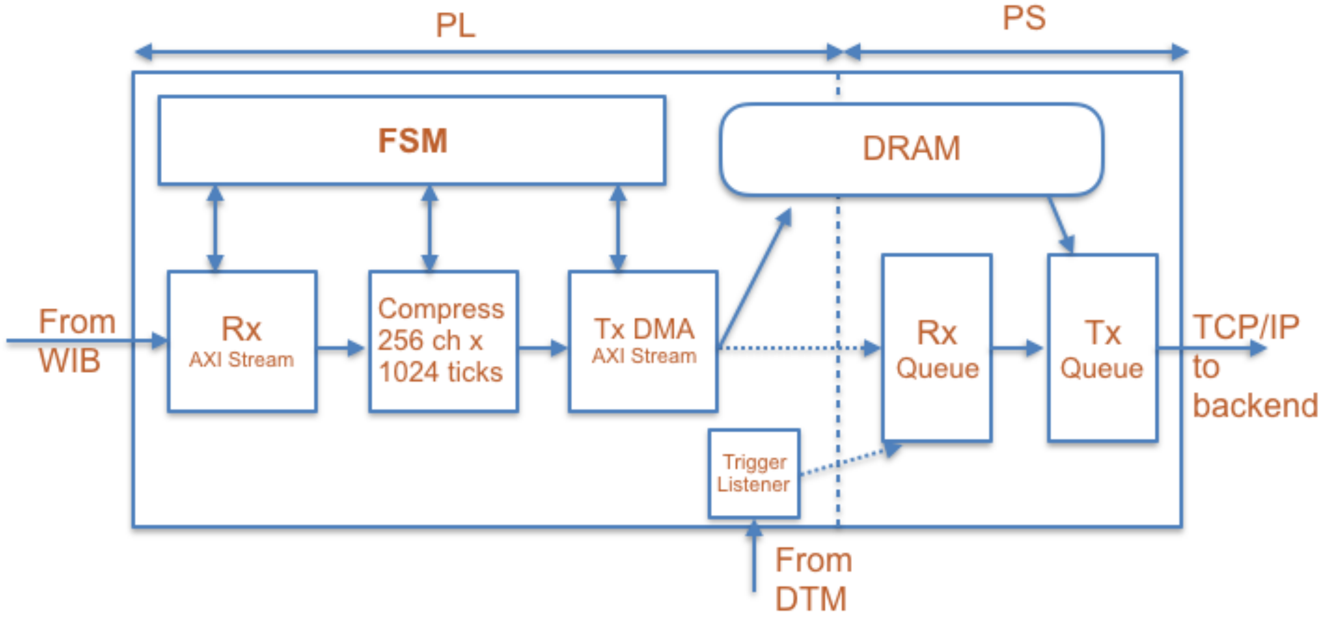

Figure 4: Schematic of the data flow through the RCE

These steps are done in a combination of the FPGA firmware and the ARM processor. A schematic of the data flow process is shown in Figure 4.

Each step through the RCE and out via the on-COB switch is a potential bandwidth bottleneck or other constraint. Below, we go through them and note the limitations.

- Each Zynq has up to 12 10-Gbps MGTs connecting to the FPGA fabric while we are using 2 of them running at ~5Gbps. This will not be an issue for protoDUNE.
- The Rx block has to be fast enough to keep up with the 2 MHz of data coming in; this has already been demonstrated on the 35ton.
- The compression block must also satisfy timing constraints (1024 x 2MHz) as well as resource constrains (DSPs & memory blocks). The compression is done wire-by-wire and is mostly parallelized. While the compression firmware is not entirely finished, from what we've seen we should be well within the timing and resource constraints for 256 channels/RCE.
- The total bandwidth between the FPGA fabric and the DRAM is specced at 10 Gbps though there is some overhead depending on what the processor is doing (as well as the design of the DMA engine). With the current design, we are seeing ~7.5 Gbps which is just a bit under the full, uncompressed data bandwidth. However even a small compression factor gets us well under the memory bandwidth limit.

- There is (conservatively) ***~500 MB*** of free DRAM to use as buffer, the rest being used by the system. This number and the ethernet bandwidth below work together to limit our maximum trigger rates.

	Steady State	Max Beam-On	100 Hz + Max Cosmics	Improved Bandwidth	Noisy
Compression Factor	4	4	4	4	2
Time-per-trigger	5ms	5ms	5ms	5ms	5ms
RCE Memory Buffer	500 MB	500 MB	500 MB	500 MB	500 MB
RCE-out rate (MBps)	50 MBps	50 MBps	50 MBps	100 MBps	50 MBps
Beam-on rate	45 Hz	140 Hz	100 Hz	200 Hz***	65 Hz
Beam-off rate	45 Hz	0 Hz	30 Hz	75 Hz	5 Hz
Max latency to Event Builder	~0 s	~ 10 s	~ 6s	~4 s	~ 10 s

Table II: Some protoDUNE TPC readout scenarios. Details are provided in the text.

- The current settings for the RCEs connect to the on-COB switch using 1 GbE however using the ARM archLinux TCP/IP stack we are only seeing ***~50 MBps*** throughput. This is actually limited by the ARM processor so there is ongoing work to perform some of the work in hardware. Eventually (next few months) we will implement 10 Gbps ethernet links between the RCEs and switch.
- The on-COB switch has a single 10 Gbps port to the outside. Currently this is not limiting but will be once we go to 10 Gbps RCE-switch links.

With these constraints in mind, we can estimate the maximum throughput and trigger rates we can handle for protoDUNE. A few scenarios are summarized in Table II and described below. In all scenarios, we assume a beam-on/off spill cycle of 4.8/19.2s and maximize the rates such that (1) the memory buffer on the RCE does not fill up and (2) the memory buffer is completely drained by the end of the spill cycle. The scenarios in Table II:

1. Steady State: here we ignore the spill structure and simply calculate the average trigger rate at which we can keep up without filling the buffer. This isn't a likely run mode, but is still an interesting scenario.
2. Max Beam-on: this maximizes the trigger rate we take with the beam on, fully filling the RCE memory buffer and then draining it in the inter-spill period. Note that there is some leeway to take a few Hz of inter-spill triggers.
3. 100 Hz+max comics: maximizes the cosmics rate taken during the inter-spill period, with a 100 Hz trigger rate while the beam is on.
4. Improved Bandwidth: assumes we can get the RCE ethernet bandwidth up to 100 MBps. Note that 200Hz while saving 5ms/trigger is 100% livetime.

5. Noisy: assumes we can only achieve a compression factory of 2 (which is equivalent to ~6 bits of noise).

The protoDUNE trigger rate requirement is ***25Hz***, so the above numbers show we should have plenty of headroom.

4.2 Timing and Trigger Distribution

The RCE will need to append a global timestamp and receive and apply the trigger to the incoming data. The timing system sends the timestamps and triggers encoded on the base clock which each COB receives at the RTM. We then must decode these messages and distribute them on the COB to each RCE. We do this by using a simple CDR chip on the RTM and then send the separate clock and data streams to the DTM which then fans out the streams to the RCEs (this is what the DTM is designed to do). A schematic of this is shown in Figure 5. A picture of the protoDUNE RTM (v1) is shown in Figure 6.

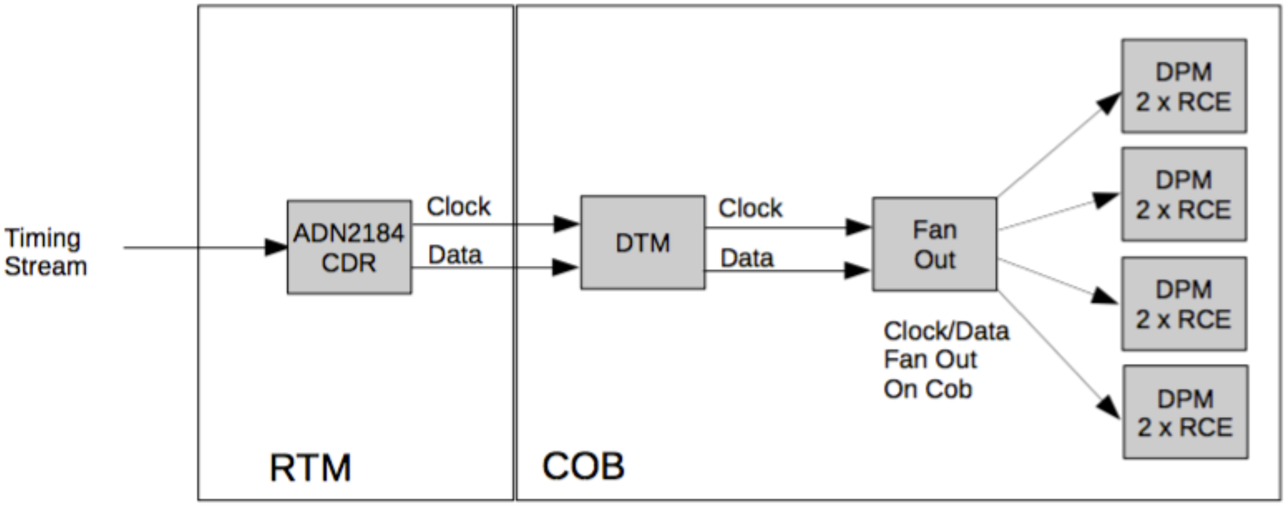

Figure 5: Schematic of clock and trigger/timestamp data distribution on the COB

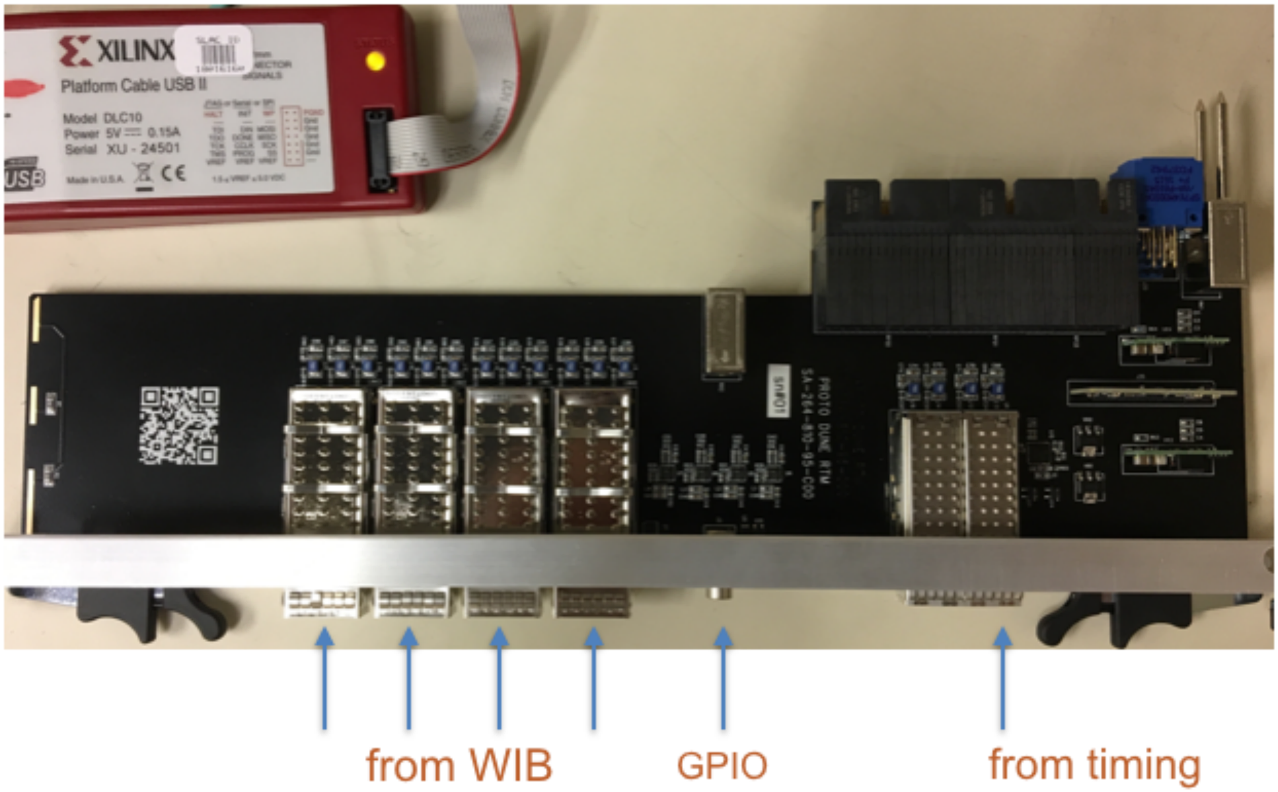

Figure 6: Picture of the protoDUNE RTM (V1).

Appendix

A. Table of resources

The resources needed for the ProtoDUNE warm TPC readout (6 APAs) are given in table below.

Items	Units	On hand as of Nov. 2017	Notes
“RCE Module” = COB+4 DPM +DTM	8	3 (BU, CERN, SLAC)	
RTM	8	3	
Optical Transceivers: Flange-to-RCE	60	~20	

aTCA Shelf (>= 8 slot); shelf manager & fans	1	1 (at CERN)	14-slots are standard;
Optical Fiber Bundles	60	0	
SFP+ Transceivers (RCE-to-PC)	16	~3	
SFP Fiber Bundle	8	~3	

B. Details of the team

Table 2 gives a list of team members, their expertise and the amount of time they are available for the ProtoDUNE work. Working document, will evolve as possible consortium is forming.

Some overlap between the RCE/COB provision and the overall installation, commissioning and running of ProtoDUNE, this overlap is indicated in the table in separate columns.

Institution	Individual	Contribution(s)	FTE Estimate	
			On RCE project	CERN Install/run general work
UC-Davis	Jingo Wang	RCE Application Software/Firmware	0.1	0.8
FNAL	Kurt Biery	artDAQ Interface	As needed	As needed
FNAL	John Freeman	artDAQ Interface	As needed	As needed
Oxford	Babak Abi	RCE Application Software/Firmware	0.4	0.1
Oxford	Farrukh Azfar	RCE Application Software/Firmware	0.3	0.3
Oxford	Justo Martin-Albo	artDAQ Interface	0.3	0.2
RAL	Tim Nichols	artDAQ Interface	0.3	0.0
SLAC	Matt Graham	RCE Application Software	0.4	0.1
SLAC	JJ Russell	RCE Application Software/Firmware	0.5	0.1
SLAC	Ryan Herbst	RCE Base Software/ Firmware & Warm Interface (RTM)	0.25	0.0

SLAC	Larry Ruckman	RCE Base Software/ Firmware	0.25	0.1
SLAC	TBD Postdoc	RCE Application Software/Firmware & RCE DAQ Operations	0.1	0.8