Presented at IEEE 17th Real Time Conference, Lisboa, Portugal, 24-28 May 2010.

The Back-end Electronics of the Time Projection Chambers in the T2K Experiment

D. Calvet, I. Mandjavidze, B. Andrieu, O. Le Dortz, D. Terront, A. Vallereau, C. Gutjahr, K. Mizouchi, C. Ohlmann, F. Sanchez

***Abstract*—Among other detectors, the T2K neutrino experiment comprises three large time projection chambers segmented into over 124.000 electronics channels. The back-end electronics system is designed to distribute a reference clock to the front-end electronics, aggregate event data over seventy-two 2 Gbps optical links and format events that are sent via a standard PC to the global data acquisition system of the experiment. The core of this system is a set of 18 Data Concentrator Cards based on an inexpensive commercial Field Programmable Gate Array evaluation kit with specific add-ons. We describe the adaptations that were made to the original platform, and detail the design of the firmware and software running on the embedded PowerPC processor of the FPGA of a Data Concentrator Card. We show how the intrinsic parallelism and a mixed firmware and software implementation of the data reduction and acquisition tasks lead to a flexible system capable of extracting in real time meaningful information from the 2.5 GByte/s of raw event data produced by the front-end electronics at a nominal rate of 20 Hz.**

***Index Terms*—Data acquisition systems, field programmable gate arrays, event building.**

I. Introduction

THE Tokai to Kamioka (T2K) experiment is the latest generation long baseline neutrino oscillation experiment that started operation in Japan [1]. A near detector (nd280) is used to characterize the neutrino beam 280 m from the source and a massive far detector located 295 km away, under Mount Kamioka, is used to perform comparable measurements. Precise track reconstruction in the near detector is achieved by three large time projection chambers (TPCs) and two fine grain detectors (FGDs) [2]. The front-end electronics of both sub-detectors is based on the 72-channel AFTER chip [3].

Each TPC comprises 24 detector modules segmented into 1728 pads leading to a total channels count of 124.416 for the three TPCs. A detailed description of the front-end electronics can be found in [4]. The data of each module are multiplexed over a single 2 Gbps optical fiber. The seventy-two 2 Gbps optical fibers of the complete system are connected to a set of 18 Data Concentrator Cards (DCCs).

Each DCC performs the aggregation of the data from four front-end modules. Complete TPC events are collected over a private Gigabit Ethernet switch inside a PC which is also connected to the experiment-wide local network. The architecture of the TPC readout system is shown in Fig. 1. Services shared with other detectors are provided by the Midas framework [5]: on-line database, global event builder, run-control and logger to mass storage. The experiment-wide clock and trigger are provided by the Master Clock Module (MCM). These signals are forwarded to each detector within T2K nd280 through a Slave Clock Module (SCM). The MCM and SCM are identical hardware boards designed by UK collaborator groups.

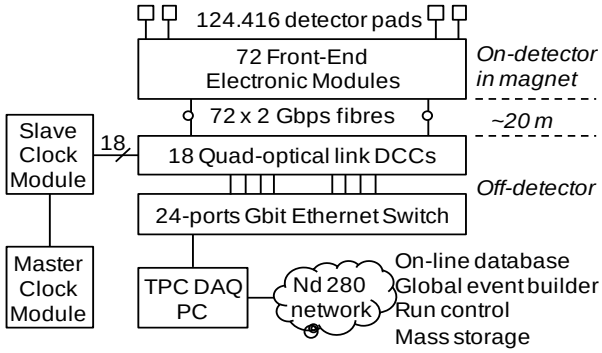

Fig. 1 . TPC readout architecture.

The present paper details the design of the DCCs and the TPC data acquisition system. This system is challenging in a number of aspects: the interface to the front-end electronics uses a 144 Gbps aggregate bandwidth optical data path; clock and trigger information as well as configuration data have to be transported over these optical links in the opposite direction. An 18-node to one-node Gigabit Ethernet switch-based event builder is needed, and integration within the global framework of the experiment has to be devised. The required event acquisition rate is 20 Hz. Raw events (120 MB of data) are brought to a manageable size by the front-end electronics and neither data processing nor throughput set a high demand on TPC back-end electronics. Because noise and channel occupation are extremely low in the TPCs - typical

Manuscript received June 2, 2010.

D. Calvet and I. Mandjavidze are with Institut de Recherche sur les lois Fondamentales de l'Univers, CEA Saclay, F-91191 Gif sur Yvette, France (e-mail: denis.calvet@cea.fr, irakli.mandjavidze@cea.fr).

B. Andrieu, O. Le Dortz, D. Terront and A. Vallereau are with Laboratoire de Physique Nucléaire et des Hautes Energies, IN2P3, 4 place Jussieu, 75252 Paris, France (e-mail : Bernard.Andrieu@lpnhep.in2p3.fr, ledortz@in2p3.fr, Diego.Terront@lpnhep.in2p3.fr, Alain.Vallereau@lpnhep.in2p3.fr)

C. Gutjahr, K. Mizouchi and C. Ohlmann are with Triumf, 4004 Wesbrook Mall, Vancouver, British Columbia, V6T 2A3, Canada (e-mail: curtis.gutjahr@gmail.com, cohlmann@triumf.ca, kentaro@triumf.ca)

F. Sanchez is with Institut de Fisica d'Altes Energies, Barcelona, Spain (e-mail: Federico.Sanchez@ifae.es)

zero-suppressed events are ~70 kB - the final throughput is only ~2.5 MB/s at the target event rate.

Besides technical aspects, limited time was a real constraint with only five years from the definition of the front-end ASIC to the full installation of a complete system ready for physics data taking.

II. Hardware Design of the DCCs

A custom board was originally planned for the readout of the TPCs and FGDs in T2K, but progress on the prototype were too slow to meet project deadlines. To cut design time and minimize the technical risks associated to the conception and production of a complex FPGA board, we decided to use commercially available evaluation boards and customize them to our needs as explained in the proof-of-principle described in [4]. We chose the Xilinx ML405 development board [6] for several of its attractive features: a FPGA with an embedded processor (Virtex 4 – PowerPC 405), 128 MB of SDRAM, a Gigabit Ethernet port, RocketIO transceivers, and more. Nonetheless, this platform has some limitations. Only four of the eight RocketIO transceivers of the FPGA are available to the user (one is routed to an optical module, one to SMA connectors, and two to SATA connectors). The clocking circuitry is not sufficiently flexible for our application: the four RocketIO transceivers cannot use a common and externally supplied reference clock.

Besides technical features, additional benefits come from the fact that the design of the ML405 board is proven, well documented, the board is very cheap and available worldwide. On the other hand, the form factor is non-standard and evaluation cards are normally not designed for building a production system where boards are densely mounted inside a rack, get power from a backplane, and allow clean cable management via a front or rear I/O panel. We detail below how these various limitations were alleviated.

A. Optical Extension Card

Our optical extension card is designed to convert the electrical interface of three RocketIO transceivers of the ML405 board to optical ports. This board is very simple and only contains three cages for Small Form-factor Pluggable (SFP) optical transceivers, SMA and SATA connectors to interface to the ML405 board, a voltage regulator and passive components. The card is powered from the ML405 board. A picture of the optical extension card and clock extension card (see later) mounted on their ML405 board is shown in Fig. 2.

B. Global Clock and Trigger Fanout

Our application requires all front-end electronics be synchronized with a global, experiment-wide, 100 MHz reference clock. In the front-end, the reference clock is derived from the clock recovered by the receiver side of the RocketIO connected to a DCC-port over an optical link. In order to ensure the proper distribution of the global clock, all the RocketIO transmitters of a DCC and all transmitters across all DCCs have to be fed with the same reference clock.

At the inter DCC level, a Slave Clock Module is used to fanout the global clock and trigger information to the 18 DCCs of the TPCs. Each DCC is connected to the SCM by a standard RJ45 cable in a star topology. Three of the four pairs of each cable are used: one pair transports the free running 100 MHz reference clock, a second pair carries the serially encoded trigger information, and the third pair is used to transport information from the DCC to the SCM: trigger acknowledge messages and rate throttle. The DCC-SCM links use Low Voltage Differential Signaling (LVDS) and are DC-coupled. A short (1 m), good quality cable (category 6) is essential to preserve clock purity.

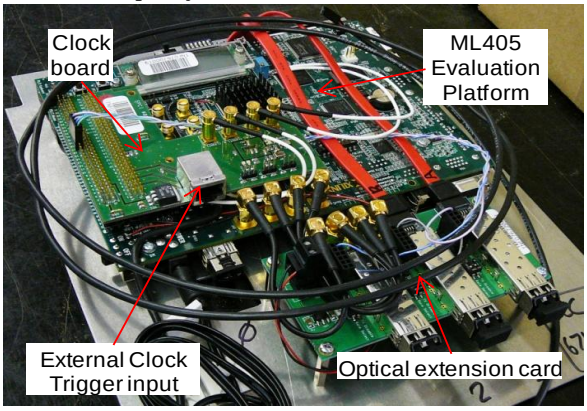

Fig. 2 . A Xilinx ML405 board with extensions makes a DCC.

At the intra-DCC level, the distribution of an externally provided clock hits several limitations of the ML405 board. Firstly, the reference clock has to be fed to both sides of the FPGA because the internal routing of the device is inadequate to carry a clock with the level of jitter required by RocketIO transceivers. Secondly, the limited range of the internal PLL of the RocketIO requires the reference clock be 200 MHz for operation at our design rate (2 Gbps over the fibre, i.e. 16 bit at 100 MHz before 8B-10B encoding). Lastly, the reference clock for RocketIO's have tight jitter requirements, though clocking circuitry cannot easily be changed on the ML405.

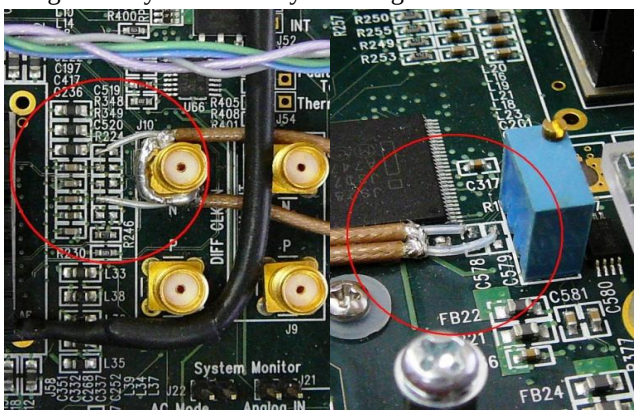

Fig. 3 . Modified reference clock inputs on a ML405 board.

We designed a clock and trigger receiver daughter card for the ML405 board and we removed some of the original clock components to be able to feed to this platform our externally supplied reference clock. A RJ45 connector is used for the connection to the SCM. The reference clock is fed to a Silicon

Laboratories Si5326 PLL for jitter attenuation and clock fanout. For standalone operation, the reference clock can also be provided by a local oscillator. On the ML405 board, the four capacitors coupling the original oscillators to the two differential reference input clocks of the FPGA are removed. Manually soldered short cables bring the reference clock delivered by our daughter card. This adaptation, shown in Fig. 3, obviously deviates from manufacturer's recommendations but in practice, operation is remarkably stable.

C. Mechanical Integration

Each ML405 board with its add-on cards is mounted on an aluminum plate. Mechanical parts of commercial crates were used to build a custom crate capable of housing up to six ML405-based DCCs as shown in Fig. 4.

Fig. 4 . Rear view of a custom-made crate populated with its six DCCs.

The optical fibers, power supply input and Ethernet cable of the DCCs are at the back of the crate. The CompactFlash, JTAG connector, and configuration switches of each DCC are accessible at the front. Not all LED's are clearly visible and the RS-232 connector of the ML405 is no longer accessible, but these are minor limitations. Our mechanical structure is not as dense as what is achieved with usual standards (21 boards per crate), but sufficient rack space was available and this arrangement well matches the segmentation of our detectors: each crate of six DCCs reads out 24 detector modules, i.e. exactly one TPC (or one FGD). Three crates of DCCs are needed for the TPCs, and two crates are required for the FGDs. In total there are 30 DCCs in operation within T2K and all of them are based on the design described above.

III. Firmware and Software for the DCC

The model used for the data acquisition of the TPCs in T2K follows the well-known client/server paradigm. This concept is implemented at several stages within the system: the DCCs are a client for the front-end electronics which acts as a server of detector data; each DCC is a server of event data for the client PC that interrogates them. The central software element of the DCC is a command server program which receives orders from the TPC data acquisition PC over an Ethernet connection, decodes, translates and posts the corresponding commands to the front-end electronics over its optical communication links, receives the responses from the front-end, encapsulate them in Ethernet frames, and returns this information to the client PC.

A. Command Server

For easier debugging and versatility, the commands accepted by the DCC command server follow a simple pre-defined syntax and are human-readable lines of plain text. The user can input commands directly from keyboard for elementary debugging. Scripts are used to perform complex tests. In the final system, a program running on the client PC connected to the DCCs translates the actions required to control data acquisition into the appropriate sequence of ASCII commands interpreted by the DCCs. In T2K, the "tpcfedcc" program is the interface between the Midas data acquisition system used by the experiment and the command interpreter of the DCCs. The easy-to-understand plain-text command approach provides a uniform way to control this electronics and several small detector R&D projects are successfully exploiting the front-end electronics and the DCC originally devised for T2K.

Most commands are self-explanatory, e.g. "sca start" and "trigwait" are used to start signal sampling in the Switched Capacitor Array (SCA) of the front-end ASICs and wait for an external trigger respectively. More sophisticated commands allow several operations at once, e.g. "thr 0:2 * 3:78 0x10" sets to 16 (0x10 in hexadecimal) the threshold of channels 3 to 78 of all ASICs on front-end cards #0 to #2 of the currently selected detector module. All configurations commands are answered by one response in ASCII format. A request for detector data can trigger a variable number of response frames. For performance reasons, frames that contain detector data are sent in binary format.

The command server program runs on the PowerPC 405 processor embedded in the Virtex 4 FPGA of the DCC. It is a single task, single user program that processes requests received over the Ethernet port in a strict first-arrived first-served basis, and with a best effort response time. This application does not require a multi-task operating system and, although porting the command server software to Linux was done, the version used in T2K is standalone. A minimal, zero-copy version of UDP/IP was implemented for communication over Ethernet with the client PC. Dedicated code based on Xilinx FAT file system library was developed to upgrade in-situ the DCC firmware and software stored on the CompactFlash card of each ML405 board. This feature is extremely useful for remote maintenance and avoids the painful task of manual memory card replacement by an operator on-site.

B. Firmware and Data Acquisition Protocol

The command server of the DCC was initially developed for the test bench of the AFTER chip: for this application, requesting the data of each channel sequentially was sufficient. As demand for a more sophisticated system grew, improvements were made: testing all detector modules individually was done with a version of the command server

where all the data of one AFTER chip (76 channels x 511 time bins x 12 bit per sample) can be requested at once. However, the sequential nature of this scheme is inadequate to meet the 50 ms response time aimed for the final T2K experiment where each DCC reads out 96 AFTER chips (i.e. 6912 detector pads plus 384 non-instrumented channels). To overcome the limitations of the trivial protocols used in the intermediate steps of the project, a more sophisticated method was devised. This scheme is described below.

The fast protocol still follows the client/server model, and the acquisition of the data of one event is initiated by the DAQ PC via a command sent over Ethernet to each DCC. This command simply instructs each DCC to return the data of the next event, and it specifies the acceptable fragment size (e.g. 64 KB per request). If the end of the event is not reached after the allowable amount of data has been returned, the client PC issues sub-sequent requests. This scheme is adequate to guarantee proper flow control over UDP/IP and avoids data being lost by overflowing network elements. Upon reception of the order to transfer event data, the DCC initiates the following actions:

- If enough buffer space is available to accept the next slice of front-end data, it multi-casts to its front-end modules over the optical links the current channel request and prepares the next request,
- It unloads the data received from each front-end module for the previous request (if any), checks the event number and timestamp, and appends the block of data to the Ethernet frame under construction,
- If enough data was accumulated to fill a frame, it sends it over the Ethernet port to the client PC.

The previous operations are repeated without intervention of the client PC until all channels have been readout (only 1896 iterations are needed because all four front-end modules are readout in parallel), or the allowable data size specified by the client has been reached: in this case the DCC waits for the next request from the client PC to pursue the transfer of the event.

The block diagram of the functions implemented in the FPGA of a DCC is shown in Fig. 5.

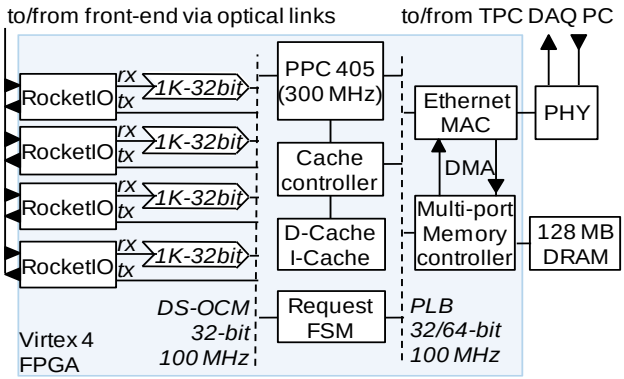

Fig. 5 . Internal functional blocks of the FPGA of a DCC.

We use many hard Intellectual Property blocks embedded in the Virtex 4 FPGA: PowerPC processor, tri-mode Ethernet MAC, RocketIO transceivers, FIFOs, etc. As will be justified later in this paper, we use two different busses to connect peripherals to the processor core: the Processor Local Bus (PLB) is used for the multi-port memory controller and the Data Side On-Chip Memory (DS-OCM) bus [7] is used for the interface to the RocketIO transceivers.

The front-end modules are capable of sending data much faster than the DCC can unload the received data, encapsulate the relevant part in datagrams and send them over Ethernet. Flow control between the front-end modules and the DCC is therefore mandatory. The maximum packet size that a front-end module can send to its DCC at once is 1044 bytes (packet header followed by the 511 twelve bit ADC samples of one channel coded on two bytes each). The DCC receives data from the front-end in an array of four 1023 word deep x 32-bit wide FIFOs connected within the FPGA to four RocketIO transceivers. At worst, each FIFO can store the data of three channels. The DCC is only allowed to post the next data request to its front-end if all receiving FIFOs have at least 1044 bytes of available space. The threshold is set to 2 KB for easier translation into the hardware.

We made two implementations of the proposed data acquisition protocol: one relies entirely on the embedded PowerPC 405 processor to post requests to the front-end, unload the received data and prepare the frames to be sent over Ethernet to the client, the second approach uses a hard-wired Finite State Machine (FSM) running concurrently with the processor to post data requests to the front-end. This is shown in Fig. 6.

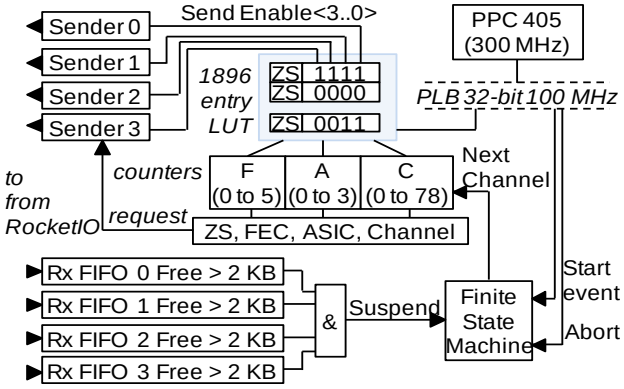

Fig. 6 . Finite state machine for front-end data gathering.

In the later implementation, the processor is still in charge of unloading the data received from the front-end FIFOs and building the Ethernet frames to be sent. In both implementations, a 1896-entry x 5-bit Look-up Table (LUT) - built from a 2 KB x 9-bit embedded dual-ported SRAM - is scanned to determine the subset of front-end modules to involve in the current channel data request. Normally all four modules controlled by a DCC are concerned by all requests if no front-end element is missing or malfunctioning. The request specifies to the front-end modules from which Front-End Card (FEC), which ASIC, and which channel digitized data has to be fetched. A per-channel zero suppression flag (ZS) allows reading some channels in uncompressed mode (e.g. non-instrumented channels which are useful to obtain the full

baseline waveform) while channels connected to real detector pads are readout with zero-suppression. A global ZS flag is also available to override the per-channel setting and force readout in uncompressed mode. Forcing uncompressed readout mode is used in T2K for pedestal monitoring and laser calibration events which are interspaced with beam and cosmic events readout in zero-suppressed mode.

IV. TPC DAQ PC and Back-End Software

The event data from the TPC goes through several logical stages after the concentration made by the DCCs. The different processes are mapped to physically distinct PCs as sketched in Fig. 7.

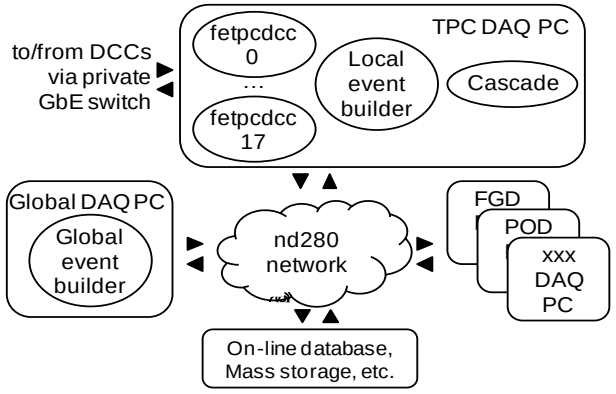

Fig. 7 . TPC back-end DAQ in the overall nd280 DAQ.

The TPC DAQ PC runs one "fetpcdcc" process for each of the 18 DCCs that are controlled. The "fetpcdcc" program is a Midas front-end program that controls configuration and data acquisition of one DCC. Duplicating identical processes rather than controlling several DCCs from a common program simplified software design at the expense of a larger memory footprint compared to a multi-threaded application. The proposed scheme naturally exploits the parallelism of a multi-core computer although it also runs on a single-processor platform without any change to the code. Data from all "fetpcddc" programs are gathered within the TPC DAQ PC with a local event builder provided by the Midas framework. A program called "Cascade" allows the interconnection of several Midas-based event builders in a global event builder as detailed in [8]. The complete description of the global nd280 data acquisition system is beyond the scope of this document. Refer to [9] for details.

V. System Performance Assessment

A. Clock and Trigger Global Synchronization

The method used to fanout the synchronization clock and trigger signals to the TPC front-end electronics allows limited precision in the control on the skew of these signals. This comes from the fact that the clock recovered by RocketIO transceivers in Xilinx Virtex 2 Pro (used in the front-end) and Virtex 4 devices (used in the DCCs) does not feature a fixed phase offset with respect to the clock of the transmitter from one system reset to the next. Some techniques have been proposed to overcome this limitation [10], but the requirements for the TPCs in T2K are compatible with the performance achieved without these optimizations. Detector pads are sampled with a 40 ns period, and a skew of up to four clock cycles in the distribution of the 100 MHz primary clock and trigger signals does not seriously impact performance. After power-up, the phase offset between a transmitter and its receiver remains constant during operation. Calibration constants can be extracted from the data and remain valid until one or several links are re-configured, which is not frequent. To study this aspect, we performed manually a series of 40 re-configurations of a DCC and we measured the relative skew between the trigger signals at the level of two front-end modules. The extreme values of the measured skew are +26 ns and -14 ns (i.e. a 40 ns peak-to-peak spread). In 75% cases the absolute value of the skew is below 10 ns, and it is over 20 ns for 7% of these trials. The skew observed partly comes from that introduced by the receiver on the front-end side (+-10 ns), and that due to the DCC. For example, the reference clock fed to the RocketIO is 200 MHz and the division by a factor two to obtain the desired 100 MHz transmit rate introduces an un-controlled skew between transmitter pairs because the division is made by distinct uncorrelated PLLs inside the FPGA.

Although the present clock and trigger fanout procedure is adequate to our needs, designers of similar systems that have tighter skew tolerance need to understand the intrinsic limitations of embedded transceivers in FPGA and how sub-optimal features can be improved.

B. Optical Link Communication Robustness

A very critical aspect in the present system is the ability of the TPC back-end electronics to establish reliable communication with the front-end over the seventy-two duplex 2 Gbps optical fibers deployed. For simplicity and performance reasons, only elementary error checking is made at both ends of these links. Re-transmitting data in case of errors is not practical. Link robustness is particularly delicate because the distribution of the reference clock to the RocketIO transceivers of the DCCs had to be customized in a way that cannot follow recommended design practices. We monitor various errors on all optical links: 8B-10B disparity errors, invalid characters, characters not in table, cyclic-redundancy check (CRC32) errors (front-end to back-end link), and parity errors (link in the back-end to front-end direction).

Most of the time, all links are very stable and no error are detected over several hours. The estimated bit error rate is under ~10^{-15} at the global level. Some errors typically appear during the transient phase shortly after system power-up, and bursts of errors are sporadically observed. The data acquisition of the TPC is not always sensitive to link errors in the front-end to DCC direction because only a small fraction of the available bandwidth is used (~0.1% for zero-suppressed events, up to 10% for the complete readout of a detector module at the maximum possible speed sustainable by a DCC). Bit errors in the DCC to front-end direction are not critical as long as the flip of an encoded bit does not cause the apparition

of a spurious trigger. The rate of trigger words is extremely low compared to the DCC to front-end electronics data rate (20 Hz compared to 100 MHz), and the risk of corruption of trigger words is negligible.

C. Data Acquisition Throughput

We measure the maximum event rate that can be captured by a DCC to readout one detector module, i.e. up to six Front-End Cards, in zero-suppressed mode, for different settings of the depth of the SCA buffer in the front-end. Thresholds are set to a sufficiently high value to produce minimal size events.

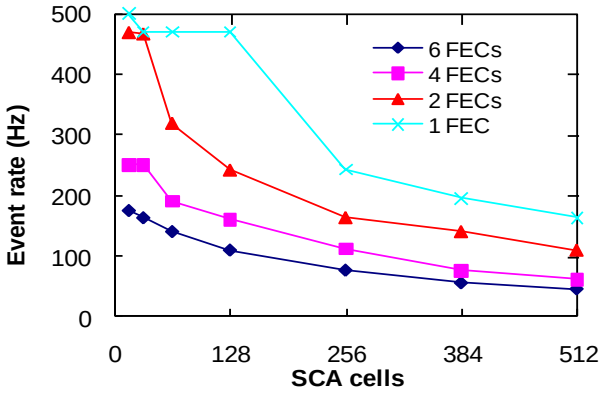

Fig. 8. Event acquisition rate of one front-end module in zero-suppressed mode.

The sustained achievable event rate in these conditions is plotted in Fig. 8. This graph is mostly relevant to small systems or a partially equipped detector. In the actual T2K experiment, each DCC reads out four front-end modules and exploits the full 511 time-bins depth of the AFTER chip. The event rate measured in experimental conditions with beam or cosmic events is ~40 Hz with a single DCC and is reduced to ~30 Hz for the complete system (18 DCCs). The distribution of dead-time is very narrow, but from time to time very large events occur and need to be truncated to meet the 50 ms response time aimed for in T2K. The upper limit on the event taking rate is mostly determined by the time needed to retrieve data in the frontend and apply zero suppression on-the-fly.

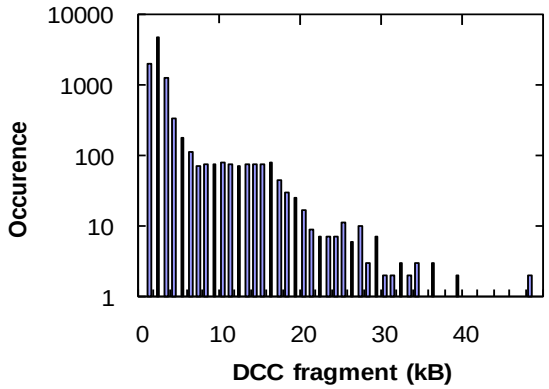

Fig. 9. Distribution of data size per DCC combined for beam and cosmic events.

Event gathering time does not increase very much with event size as long as events are relatively small and hit channels are uniformly spread across detector modules.

We show in Fig. 9 the distribution of event fragment size per DCC for beam and cosmic events. With a threshold set to 4.5 standard deviations above channel noise level, the mean event size is 3.5 kB per DCC.

At worst, the acquisition time reaches its maximum when full events are readout in uncompressed mode. This is shown in Fig. 10 for one DCC. Event acquisition time is mostly determined by event size and it scales linearly with the number of front-end modules and number of time-bins being readout. The sustained throughput is ~16 MB/s per DCC. Global throughput scales almost linearly with the number of DCCs being readout until saturation of the Gigabit Ethernet link of the TPC data acquisition PC is reached. Data losses are observed when full events from 12 DCCs or more are readout because some network elements cannot sustain permanent saturation. In practice, full event readout is used in T2K for pedestal events and laser events. For these, only a subset of DCCs is readout and fewer than the 511 available time-bins per channel are acquired. For laser events, data are only acquired from the DCC corresponding to the area currently illuminated by the laser. Only 51 time-bins per channel are readout and laser events have a fixed size of 756 kB. Data acquisition takes 52 ms, which is slightly above the target time-budget, but remains acceptable.

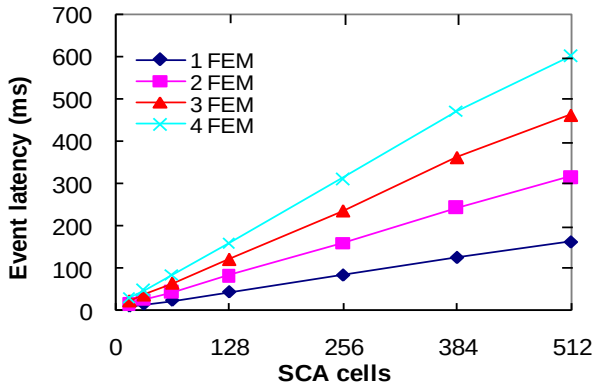

Fig. 10 . Event readout time for one DCC in full readout mode.

For large events, the throughput of a DCC is mostly determined by how quickly the embedded processor can unload data received from the front-end and encapsulate them in Ethernet UDP/IP frames. Earlier versions of the DCC firmware used the embedded Ethernet MAC in FIFO mode. Throughput was ~5 MB/s and it was brought to ~12 MB/s in DMA mode. Adaptation was made to ensure that all data transfers are aligned on 32-bit words boundaries: a standard Ethernet frame header is 14 bytes long and misaligned accesses occur if 32-bit payload words are directly appended after such header. Two empty bytes were added after the IP and UDP headers to correct for this mis-alignment. We observed that access by the processor to peripherals over the PLB is slow (~20 MB/s at 100 MHz clock rate) because only single beat transfers are supported. We use the DS-OCM bus instead of the PLB: we measure an average throughput of ~26 MB/s for reading out fresh data from a FIFO connected to this bus and storing them in the external SDRAM via the cache

controller. After these various optimizations, the sustained data acquisition rate achievable by one DCC is ~128 Mbit/s. This represents only ~12% of the available bandwidth of the Gigabit Ethernet output link. Using the Auxiliary Processor Unit (APU) of the PowerPC 405 as a fast internal I/O path instead of the DS-OCM bus was tested but this scheme did not increase throughput. We think that only a hardware-based mechanism for internal data movement within the Virtex 4 FPGA can fully exploit the capacity of the Gigabit Ethernet link.

D. Overall System Operation and Stability

During the two years of detector developments, several ML405-based DCCs have been operated in laboratory and test-beam setups. The complete TPC readout electronics in T2K has been in operation during an estimated 4000 cumulated hours from installation. No hardware failure has appeared yet, and operation is reasonably smooth so far. At the present stage of debugging, errors of imprecisely determined origin happen at the level of one every ~2.5 hours and simple error recovery code allows continuing a run without operator intervention. The event readout abort mechanism needs to be improved because very large events induce dead-time that causes ~0.4% of beam events to be missed.

VI. Conclusions

The design of the back-end electronics for the TPCs in the T2K experiment has been presented. This system synchronizes and collects data from 72 front-end electronics modules over a 144 Gbps aggregate bandwidth optical path. It is built from 18 customized commercial Xilinx ML405 evaluation platforms mounted in three custom made chassis. The results obtained show that a workable production system can be build from modern FPGA evaluation kits. The approach provides a cost effective and minimal design-effort solution at the expense of lower packing density and some sub-optimal aspects (e.g. delicate hardware modifications on the ML405 product were needed).

The complete TPC readout has been in operation for several months and the overall performance and stability of the data taking are satisfactory.

Acknowledgment

We thank P. A. Amaudruz and K. Olchansky from Triumf for the various technical contributions to the FGD that have been re-used for the TPC readout system.

References

[1] A. K. Ichikawa, “Status of the T2K long baseline neutrino oscillation Experiment”, in Proc. International Conference on Topics in Astroparticle and Underground Physics, TAUP2009, Italy, July 1st-5th, 2009.

[2] T. Lux, “A TPC for the near detector at T2K”, in Proc. 3rd Symposium on Large TPCs for Low Energy Rare Event Detection, Journal of Physics, Conference Series 65 (2007) 012018, IOP Publishing.

[3] P. Baron et al., “AFTER, an ASIC for the Readout of the Large T2K Time Projection Chambers”, IEEE Trans. Nucl. Sci., Volume: 55 N°3, June 2008, pp. 1744 – 1752.

[4] P. Baron et al., “Architecture and implementation of the front-end electronics of the time projection chambers in the T2K experiment”, IEEE Trans. Nucl. Sci., Volume: 57 N°2, April 2010, pp. 406 – 411.

[5] S. Ritt and P. Amaudruz, "The MIDAS Data Acquisition System", in Proc. IEEE 10th Real Time Conference, 1997, pp. 309-312. Midas online: https://midas.psi.ch/

[6] Xilinx ML405 Evaluation Platform User Guide, UG210 (v1.2) March 21, 2007. [Online]. Available: http://www.xilinx.com

[7] PowerPC 405 Processor Block Reference Guide, Embedded Development Kit, Xilinx user guide UG018 (v2.4) January 11, 2010. [Online]. Available: http://www.xilinx.com

[8] R. Poutissou, K. Olchanski and K. Wong, “Cascading MIDAS DAQ Systems and Using MySQL Database for Storing History Information”, Proc. 17th IEEE Real Time Conference, Lisbon, Portugal, 24-28 May 2010.

[9] M. Thorpe et al., “The T2K near Detector Data Acquisition Systems”, Proc. 17th IEEE Real Time Conference, Lisbon, Portugal, 24-28 May 2010.

[10] A. Aloisio, F. Cevenini, R. Giordano, V. Izzo, “High-Speed, Fixed-Latency Serial Links With FPGAs for Synchronous Transfers”, IEEE Trans. Nucl. Sci., Volume: 56 N°5, October 2009, pp. 2864 – 2873.