Tile Calorimeter Read Out Driver. Firmware Developments for the Final Prototype

J. Castelo

Dept. Física Atòmica, Molecular i Nuclear/IFIC/ATLAS-Tilecal Valencia Group
Universitat de València, Doctor Moliner 50, 46100 Burjassot, Spain
Jose.Castelo@ific.uv.es

Abstract

The work described here covers the firmware developments for the Read out Drivers (ROD) of the ATLAS hadronic calorimeter (Tilecal) for the Large Hadron Collider (LHC). This paper briefly explains the firmware design and also its impact in two set-ups: Laboratory and ATLAS Combined Testbeam.

I. ATLAS-TILE CALORIMETER ROD SYSTEM

A. Introduction

The Tilecal ROD is a project being developed by the Valencia Tilecal group [1]. It is designed to read out the calorimeter data and send them to ATLAS general TDAQ system with real time digital processing capabilities.

This implementation is based on last generation Complex Programmable Logic Devices (CPLD), the Field Programmable Gate Arrays (FPGA), and Digital Signal Processors (DSP). These devices are needed due to the strong system constrains in terms of input/output bandwidth (112,24Gbps and 53,6Gbps respectively) and processing power (68992 MIPS) required.

The lab tests and the installation of the ROD final prototypes in ATLAS Combined beam test will reveal very important results before production. Also it is important the integration issues and algorithm performance on the Processing Units in a scenario as close as possible to the final installation of ATLAS detector in the LHC.

B. ROD Crate

The ROD is organized in 4 partitions. One partition reads out a hadronic calorimeter barrel with independent trigger and dead-time handling. The hardware modules that belong to a partition are plugged in one crate (VME64x 9U) called ROD Crate (), thus there are 1 ROD crate per partition. A ROD Crate is equipped with 8 ROD modules, 1 Trigger and Busy Module (TBM [4]) and 1 ROD Crate Controller. The number of ROD

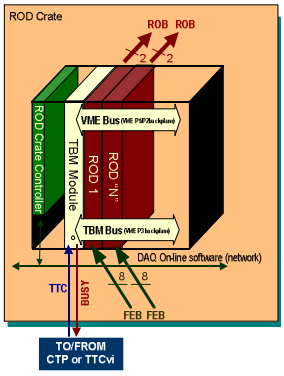

Figure 1 : ROD Crate

modules initially planned was 64, but the number of module inputs and DSP were duplicated for the final design. This double density means that the whole ROD system will be implemented in 32 VME boards with the following main interfaces: Trigger and Timing Control (TTC), ATLAS TDAQ Read Out System (ROBin), Tilecal Super-Drawers (FEB), and ROD Crate Controller (VME Single Board Computer).

C. The ROD module

The ROD motherboard (Figure 2) is a 9U VME64x board [6]. A ROD can be equipped with up to four Processing Unit boards. In ATLAS, it is foreseen to use 2 PU/board in Staging Mode (SM: 4 FEB routed to 1 PU), because the current estimation of processing power consumption demonstrates that one PU could process 196 channels with seven samples if we apply optimal filtering [8] for signal reconstruction.

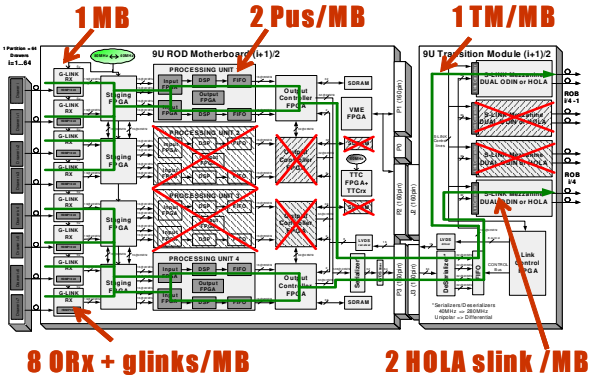

Figure 2: Schematic diagram of ROD module, TM, PU and main components of the dataflow part

This fact will decrease the cost in total project quantities, and we can maintain an upgradeable system if new power needs arise (PU is a plug-in daughter board)

The input data coming from an optical fiber is received by eight ORx with G-link chips (deserializers) and distributed to four FPGA (called Staging FPGA). The Staging FPGA (input distribution device) routes the data to the PU. In the PU, the data are treated and formatted in less than 10µs due to the big requirements of Level 1 trigger. The output data from the PU (buffered in PU FIFOs) are read by the so-called Output Controller OC FPGA (output distribution device). The OC can send the event fragment to VME or to the ROS. Furthermore, the ROD will monitor the G-links temperature and the PU output data to build online histograms.

D. The ROD Crate Controller

The ROD crate is controlled by a CPU (RCC), which performs several tasks, such as initialization of modules, monitoring, support of VME data transfer and other activities. The CPU being used now is CT VP-110 and seems to be the ATLAS standard to follow.

E. The Trigger and Busy Module (TBM)

The trigger signals coming from Central Trigger Processor (CTP) are managed by a TTC crate, and received and distributed to RODs with a TBM [4] through a dedicated optical link. The TBM distributes trigger accept and clock signals over custom P3 backplane [4]. This backplane also collects the busy signals generated by the ROD modules to the TBM.

F. The Transition module

The Transition Module boards are placed just behind each ROD module in the rear part of the crate. Each TM [4] can hold up to 4 S-Link LSC [9]. The ROD module sends the data through the backplane P2 and P3 to the TM, which transmits them via optical fibres to ROBin.

II. ROD Injector Board

The ROD Injector is a 6U sub-rack mount motherboard that provides TTC and Power supply to 2 Interface Cards (4 FEB outputs). The purpose of the card is to inject simulated data to ROD to test system at lab in absence of FEB.

The APEX FPGA is programmed with a special firmware for injecting simulated events, which are serialized and converted to an optical signal before they are sent to ROD after a trigger is issued.

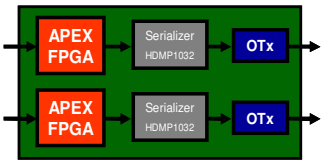

Figure 3: Interface Card

This configuration reports good control of the injected data (FPGA programming) and the possibility to test the "input interface" of the ROD with "output interface" of the real FEB hardware (Interface Card [5]).

III. The ROD Firmware

The main differences between LAr ROD [2] and Tilecal ROD are at level of firmware (FPGA and DSP devices) and software (ROD library, Graphic user interfaces and Online Software). Hardware and software differences are not the scope of this note, therefore in next chapters will be described the firmware developments for the ATLAS-Tilecal ROD [1].

In Figure 4 is shown the ROD dataflow (FEB and ROS), TTC, and VME interfaces marked in different colors. The main components and programmable ROD devices are pointed out.

The current versions of firmware tested at laboratory and ATLAS Combined Testbeam are the following:

- Firmware compatible with LAr: TM firmware [4], TTCFPGA v2.22[2], OC_FPGA v3.20[2]
- Firmware specific for Tiles: ROD Injector FPGA v1.2, FPGA PU v3.2, Staging_FPGA v4.4, DSP PU InFPGA v1.0, DSP Code v1.0

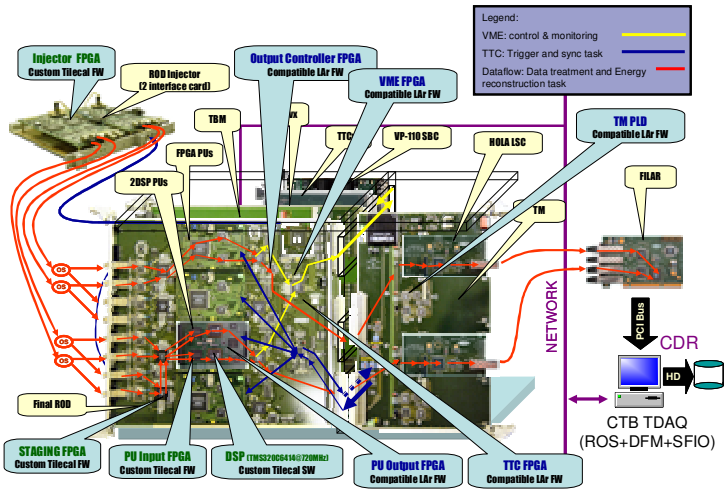

Figure 4: ROD Components, Dataflow and Firmware

All projects are designed in hierarchical VHDL. Some design tools used are: Visual Elite 3.5.1 (design entry and simulation), Leonardo Spectrum 2003 and Symplify 7.6 (for synthesis), Quartus II 4.0 (for fitting), Modelsim Altera 5.7c (simulation) and Code Composer Studio v2.21 (for DSP development IDE).

IV. ROD Injector Firmware

There are several firmware versions for injecting data depending on the features needed. All these versions include a CRC checksum at the end of the fragment for later error checking at ROD level:

A) *Send a raw data event when a L1A signal arrives*. The event injected is taken from testbeam data and stored in an internal "virtual ROM" inside the FPGA. This mode allows us to apply online algorithms because we inject "real data".

B) *Send an event with Tile's FEB format but with counter data*. The data of event "N" is one unit bigger than previous one ("N-1"). So this allows us to detect when some event fragment has not been readout in the right sequence.

C) *Send raw data with the FEB data format, but using a random energy values. These values are generated based on the shaper waveform function and a random number source*. In this mode we get the set of random samples multiplying the output of a Linear Feedback Shift Register of 10 bits by the normalized shaper waveform of the detector (see Figure 5). This allows us to simulate random FEB data for test online algorithms and ROD data format validation.

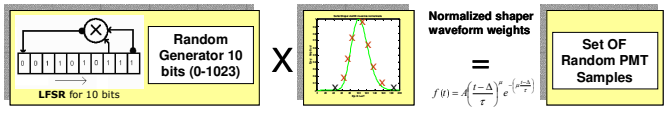

Figure 5: Random samples using LFSR and shaper f(t)

After the implementation in APEX20KE device (EP20K160EQC208-3 6400 logic elements and 81920 mem. bits) we get the results pointed out in Table 1.

Table 1: ROD Injector Firmware implementation results

Firmware Type and Version	Total Logic Elements	Total Pins	Total Memory Bits	Maximum Freq. (req. 20MHz)
A) v1.1	382 (5%)	109 (76%)	8192 (10%)	39,7MHz
B) v1.2	440 (6%)	109 (76%)	0 (0%)	37,52MHz
C) v0.9b	474 (7%)	109 (76%)	0 (0%)	38,9MHz

V. ROD MOTHERBOARD FIRMWARE

G. Staging FPGA Firmware

There are 4 Staging Chips/ROD. The Staging FPGA is the input data distributor of ROD. The main functionalities are:

- To receive deserialized data from 2 G-Links (HDMP1024) in Normal Mode (NM) or from 4 G-Links in Staging Mode (SM) and route them to a PU. At startup, it does the G-link reset and configuration.
- Clock deskew between g-link clock and PU clock with a dual-clock FIFO implemented inside this device.
- Provide VME access for configuration and status.
- Monitor G-Link Temperature and put it available to VME interface with a read only register (current, maximal and minimal values).
- Send test data from internal RAM (VME writable) to PU (debug/test purposes).

Figure 6 is a block diagram of the structured firmware implementation. The project has a deeper level of hierarchies but with this picture one can get a brief idea of the main blocks, the Input/Output and control interfaces, and the evolution over different versions (v1.x, v2.x, v3.x, etc).

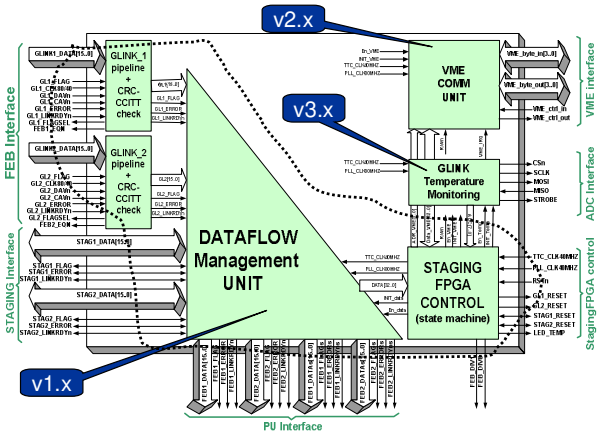

Figure 6: Staging FPGA Firmware Block Diagram

Version 1.x includes CRC-CCITT16 core and a dataflow unit. CRC core calculate CRC checksum and compare it with the one received from FEB. Dataflow Unit is complex but mainly implemented with internal dual clock fifos and SRAM memory and its control to inject VME test events. The dual clock fifos are controlled by a state machine to avoid clock deskew between G-links and Motherboard (PU) clocks. Version 2.x implements the VME core with R/W registers to configure the lots of features on this device and read g-link temperature registers. Temperature monitoring is implemented in a core (v3.x) to read SPI interface of the ADC connected to an RTD glued to g-links.

The Staging FPGA is implemented in an ACEX1k50 device (EP1K50FC484-1 2880 logic elements and 40960 memory bits). After synthesizing and fitting firmware for version 4.4 we get the next results:

- Total logic elements: 2675 (92%). For production will be mounted the bigger device ACEX1k100 to have more flexibility.
- Total Pins 197/249 (79%)
- Total Memory Bits: 33792 (82%)
- The maximum frequency in worst case is 72, 99 MHz for CLK40. This means that there is enough margin since the clock frequency is 40MHz (see Table 2).

Table 2: Staging FPGA synthesized

Clock	Required Fmax	Simulated Fmax (worst case)
CLK40	40.00 MHz (period = 25.000 ns)	72.99 MHz (period = 13.700 ns)
CLK_feb1	40.00 MHz (period = 25.000 ns)	107.53 MHz (period = 9.300 ns)
CLK_feb2	40.00 MHz (period = 25.000 ns)	116.28 MHz (period = 8.600 ns)

The design is almost final and the current stable versions are v4.4 (ATLAS CTB) and v4.4.1 (Lab set-up).

H. FPGA Processing Unit Firmware

The FPGA PU is a simple board based on one FPGA and 2 output FIFOS. The purpose of this card is to test the ROD motherboard in absence of final DSP PU and exploit the FPGA features. There are up 4 FPGA PU Daughter Boards/ROD. This board is responsible for:

- To receive FEB data from Staging FPGA (2 FEB in NM or 4 FEB in SM) in Transparent Mode. With this mode the PU sends an event fragment with raw data and ATLAS-TDAQ data format [7].
- To receive and deserialize the TTC broadcast data coming from TTC_FPGA (from TTCrx chip in MB).
- To synchronize FEB and TTC data, prepare standard data format and buffer the event fragment in FIFOs.
- To send data stored in FIFOs to the OC FPGA (4 OC in ROD MB). The OC manages formatted data and sends them to VME for monitoring purposes or to S-LINK.
- VME core for config and status registers, etc.

A block diagram of this PU and the FPGA is shown in Figure 7. In such diagram is pointed out the evolution of the different versions of the firmware. The version 1.x implements baseline features such as FEB dataflow state machine and internal buffering. This initial version was useful to use this device in standalone mode. The versions 2.x, with VME core implemented, brought a new dimension in configuration and control of the processing unit. The versions

3.x includes TTC box able to decode serial TTC lines. This parallelized TTC data is buffed into internal FIFOs to have them ready to be mounted with the right FEB fragment.

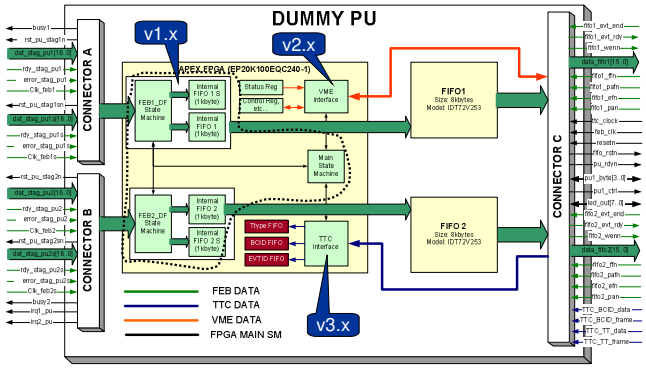

Figure 7: FPGA PU Block diagram

Again, the block diagram shows the top hierarchy of the firmware design. Most of the details remain hidden inside these boxes that will be deeper described in other dedicated notes.

The FPGA PU firmware design is based in an APEX20KE device (EP20K100EQC240-1) that has 100.000 gates, 4160 logic elements and 53248 memory bits. With the FPGA PU firmware version 3.2, the resources used in this device are:

- Total logic elements: 3954 (95%). This is a very intensive use of a programmable logic device but could be decreased because lots of features are only useful during debugging period
- Total Pins 156/183 (85%)
- Total Memory Bits: 33400 (62%)
- The maximum frequency is in worst case is 45,7MHz for "ttc_clk" (see Table 3) in a node for VME interface that doesn't have any interference with dataflow (40MHz constraint).

Table 3: FPGA PU clock simulation summary after synthesis

Clock	Required Fmax	Simulated Fmax (worst case)
TTC_clk	40.00 MHz (period = 25.000 ns)	45,7 MHz (period = 21.882 ns)
feb2_clk	40.00 MHz (period = 25.000 ns)	50.52 MHz (period = 19.796 ns)

The latest stable version used in ATLAS Combined Testbeam and LAB tests is v3.2 and the design is almost final.

I. DSP Processing Unit

The DSP PU [3] is a mezzanine (120*85 mm) board able to treat up to 96 calorimeter channels (2 FEB) in NM (4PUs/ROD) and 192 channels (4 FEB) in SM (2PUs/ROD).

The PU is equipped with two Input FPGA (InFPGA), two TMS320C6414@720MHz DSP and two Output FIFO. All these dual devices are used for getting double processing power in one PU and they are responsible for I/O dataflow and digital samples reconstruction. For control and configuration, there is an Output FPGA (OutFPGA) that implements the VME and TTC interfaces with the ROD Motherboard. See Figure 8.

The input FEB data enters the InFPGA where they are formatted and checked as needed for the DSP algorithm. When an event is ready, an interrupt is sent to the DSP which launches a DMA to read the data with the 64-bits EMIFA bus. After the DSP has finished processing an event, it writes the results in the output FIFO through the 16 bits EMIFB bus.

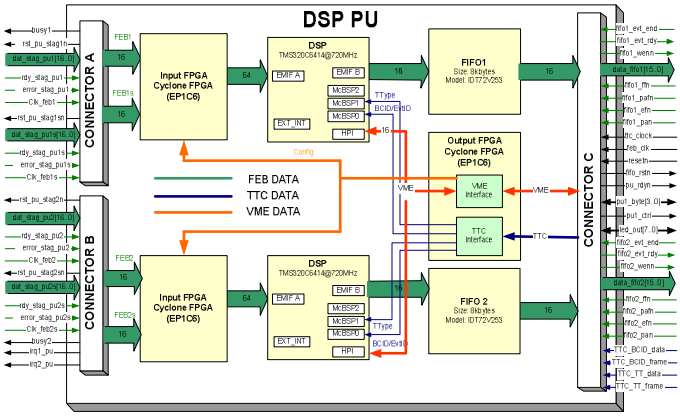

Figure 8: DSP PU Block diagram

The TTC data are received in the OutFPGA and sent to each DSP via 2 serial ports. A serial port is for the Trigger type (McBSP1) and the other is for the BCID and EventID (McBSP0).

The OutFPGA allows the control of the PU board through VME layer. Main responsibilities of this device are booting DSP code and InFPGA. After these devices are up, the OutFPGA should provide an interface with InFPGA registers and DSP ports (HPI and McBSP2 for histograms download and DSP commands respectively).

The DSP PU adds the following functionality:

- Error detection and staging mode implementation at 100KHz L1A (Input FPGA)
- Data Processing with online reconstruction algorithms for working at 100Khz L1A trigger rate (DSP). The DSP clock cycle is 1,38ns and it has 8 parallel and independent Arithmetic and Logic Units (ALU).

1) Input FPGA Firmware

There are two Input FPGA chips per PU. The main task of InFPGA is to receive deserialized data from 1 G-Link in NM or from 2 G-Links in SM and route them to DSP. To do this task we need to implement:

- VME interface for status (R) and control registers (R/W).
- Event sampler: to detect start of event.
- Internal Dual Port Memory (DPM) with two memory blocks to store previous event and current event. The dual port memory could be read and write simultaneously from two ports (Port A and B). While current event is stored addressing port A of the DPM, the other memory bank (previous event) is being read by DSP in FIFO mode with a DMA. This FIFO mode is done inside Input FPGA increasing Port B addresses after DSP asserts chip enable signal (CEn). The DSP launches a DMA when the

Input FPGA sends an IRQ to notice that an event is stored and ready to be read in DPM. In Figure 9 is simulated this behavior with InFPGA VHDL code.

- Staging Mode: to implement it we need 2 DPM/InFPGA.
- DSP interface: Read DPM as FIFO and manages CEn and IRQ signals for DMA treatment.
- Arrangements over FEB data format before writing it to DPM to help DSP in their calculations.

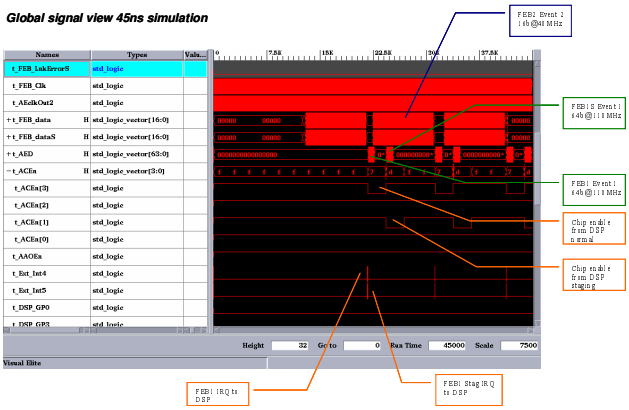

Figure 9: Input FPGA firmware functional simulation

The version 1.0 is stable and working at lab. It has the basic features needed to send raw data to DSP. The version 2.x is under development and will do online error checking (CRC16 and parity) and data format arrangements to speed up DSP processing (see Figure 10).

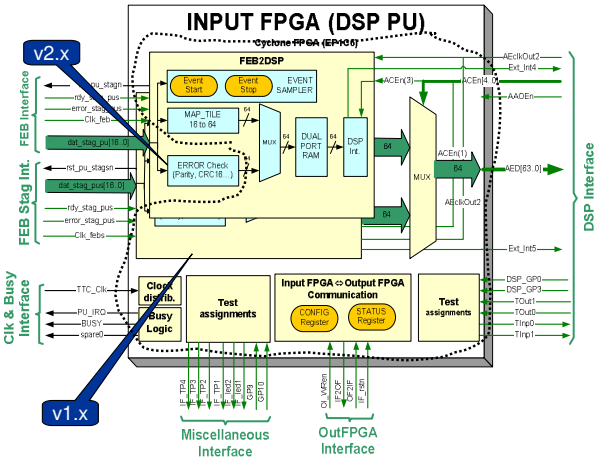

Figure 10: Input FPGA firmware block diagram

After synthesizing and fitting the version 1.0 of the firmware design for the Cyclone FPGA (EP1C6F256C8 has 5980 logic elements and 92160 memory bits), the resources used in this device are:

- Total logic elements: 513 (8%)
- Total Pins 135/185 (72%)
- Total Memory Bits: 65632 (71%)
- The maximum frequency in worst case is 196,97MHz for AEclkOut2 (DMA DSP clk) which is much bigger than the 120Mhz required. See Table 4 for more details.

Table 4: InFPGA firmware clock simulation summary

Clock	Required Fmax	Simulated Fmax (worst case)
AEclkOut2'	120.00 MHz (period = 8.333 ns)	196.97 MHz (period = 5.077 ns)
FEB_Clk'	40.00 MHz (period = 25.000 ns)	113.78 MHz (period = 8.789 ns)
TTC_Clk'	40.00 MHz (period = 25.000 ns)	155.30 MHz (period = 6.439 ns)

2) *DSP Firmware*

The DSP code is implemented for the final DUAL DSP Processing Unit based on 720MHz fixed-point TMS320C6414 of texas instruments. It is available now in the preliminary version 1.0. This code covers the main skeleton for control and configuration of the Input/Output interfaces and is written in "C" language. The Main features implemented in v1.0 are:

- Management of the interrupt based input/output DMA tasks and the processing tasks
- Builds the TDAQ standard data format.
- VME communication: for commands and download online histograms.
- Processing Task: Transparent mode (raw data copy mode).

The version with processing mode based in online reconstruction algorithms is not released in a stable version. There are some routines developed in "C" and Assembler for next algorithms: Optimal Filtering [8], Flat Filtering and Maximum sample. Some of these routines were tested during summer 2003 Tilecal Testbeam [10] in ROD Demonstrator board with DSP TMS320C6202@250MHz. Some adaptation (ASM routines) and improvements are needed to take advantage of new DSP features (cache stalls, 32 bit precision optimal filter weights, etc).

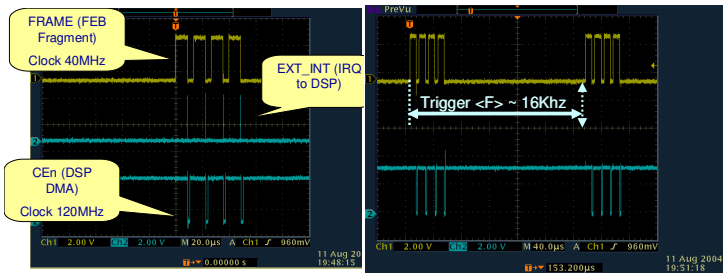

Figure 11: 4 Event group of 100Khz trigger rate

In the Figure 11 is shown two images captured with a digital oscilloscope. On the left part, the reader can see 4 events injection at 100 KHz and the protocol between the Input FPGA firmware v1.0 (IRQ to DSP) and the DSP code v1.0 (CEn, InFPGA DPM Chip Enable). Furthermore, the graphs reveals the timing characteristics watching the narrow pulse of the DMA (CEn) with 120Mhz clock and 64 bit bus and wider pulse of the FEB event from staging FPGA at 40MHz and 16 bit bus (FRAME). On the right side of the figure one can see that this PU can work at up to 100 KHz trigger rate. The mean frequency we get is 16 KHz due to the dead time (busy) generated at the end of the chain. This is because of the limited capabilities of the DAQ PC with non real-time operating system.

VI. ROD System Tests

J. Tests at Laboratory

First tests at lab revealed data corruption when injecting data with real FEB (CTB) or Injector. The suspicious source of error was the Staging FPGA firmware because:

- The ORx sensitivity was better than –17dBm (λ= 850nm) and the OTx power measured in worst case is -8,7dBm.
- The G-link Deserializer (HDMP1024) was monitored with Staging FPGA and #LINK_READY and LINK_ERROR signals did not have rising edges. This means that there are not link errors at least in C-Field.
- The deserializer reference clock jitter that could affect to BER (Random & Deterministic Jitter effects) was measured, and in all cases the clock jitter is bellow 25ps (HDMP1024 jitter budget is 468ps). See Figure 12.

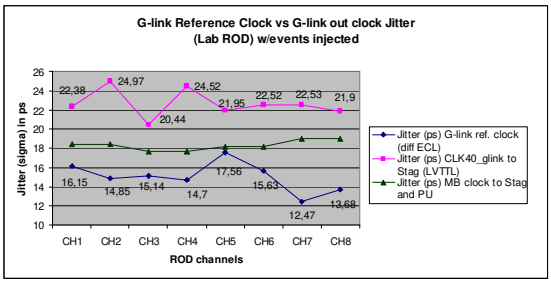

Figure 12: Clock jitter measurement

After tracing the problem we solve it introducing a dual clock FIFO controlled by a state machine to manage dead time between read and write clocks (see Figure 13)

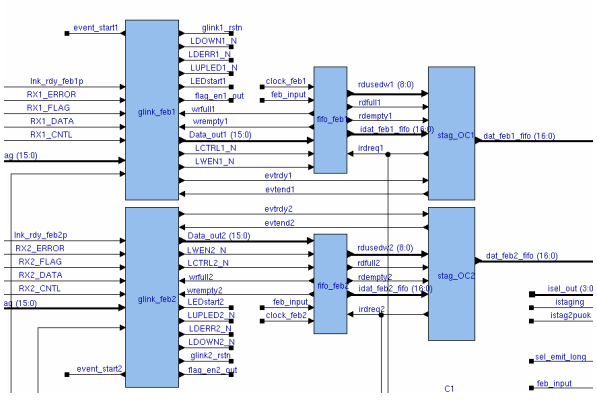

Figure 13: Staging FPGA internal dataflow block diagram

The Interface from the G-link is implemented with a 17 bit data bus synchronized with a "feb_clock". The interface with PU is a data bus of the same size but read with a free running clock in the motherboard (40MHz from QPLL or 40.08MHz from TTCrx). Due to the different clock phase and frequency shifts we get some data mismatch from time to time. A dual clock FIFO solved problem with the clock synchronization.

After this FIFO was implemented in the firmware the system was exercised with test runs of 3GEvents and obtaining zero errors. This number give us an estimation of Bit Error Rate (BER) better than $6*10^{-14}$.

K. ATLAS Combined Beam Test Installation

In the 2004 ATLAS combined testbeam (CTB) all the sub-detectors of ATLAS are configured to use the same SPS beam in H8 in order to perform a common data acquisition at the level of the TDAQ and the offline software integration. The ROD final prototype equipped with FPGA_PU was installed in CTB and the data are being acquired with stability.

VII. Conclusions

The ROD final prototype was successfully tested with custom Tilecal firmware and minor hardware modifications over the LAr ROD [2] in two set-ups: Lab and CTB. In lab, the ROD was tested with dedicated Injector Boards and in CTB was exercised with TTC and FEB stimulus.

After acquiring several long term runs (N=10) of $3*10^9$ events without errors, we can conclude that the BER is better than $6*10^{-14}$. This BER is better than expected one because the g-link deserializers (HDMP1024) have a specification of 10^{-12}. The BER is calculated over the whole acquisition chain: from data injection MB to DAQ PC. The data injected are known (counter based) and it are checked based on this fact and also validating CRC-CCITT link information.

Most of the features of the FPGA devices are implemented in stable versions. Nevertheless, future upgrades are foreseen to develop some extra features on the firmware over the hierarchical VHDL code just done.

The next step is to develop new DSP Code (v2.x) with online reconstruction and test it with real or random injected data. This is the main ongoing research line in the firmware developments for Tilecal ROD.

VIII. References

[1] Tilecal Valencia www (IFIC), http://ific.uv.es/tical/

[2] A. Blondel et al., "The ROD Mother Board for the ATLAS Liquid Argon Calorimeter", (EDMS)

[3] Julie Prast, "The TMS320C6414 DSP board", (EDMS)

[4] Pierre Matricon et al. "TBM, TM and CP3", (EDMS)

[5] K. Anderson et al. "ATLAS Tile Calorimeter Interface Card", LECC 2002, ISBN 92-9083-202-9.

[6] J.Castelo et al. "ROD General Requirements and Present Hardware Solution for the ATLAS Tile Calorimeter", LECC 2002, ISBN 92-9083-202-9

[7] C. Bee et al. "The raw event format in the ATLAS Trigger & DAQ", CERN, February 2004,

[8] F.Camarena et al. Optimal Filtering applied to 1998 Test Beam of Module 0. ATL-TILECAL-2002-015 (CERN).

[9] O. Boyle, R. McLaren, E. van der Bij, "The S-LINK interface specification," ECP division CERN, March 1997

[10] J.Castelo et al., "ROD Set-up for the TileCal Testbeam, 2003 period. Results and future plans", LECC 2003, ISBN 92-9083-216-9