FERMI NATIONAL ACCELERATOR LABORATORY

Serial Communication for Mu2e Tracker Electronics

Tommaso Vincenzi

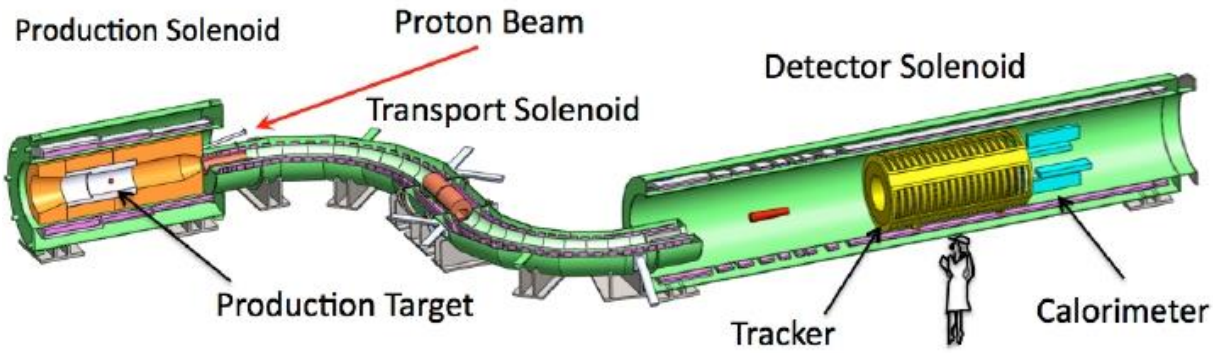
23/09/2015

This report focuses on the implementation of a fast and reliable serial communication among the electronic boards inside the Tracker of Mu2e Detector. The report is organized as follows: in the first chapter there is an introduction to the overall setup, explaining how the Tracker is organized and where the electronics is positioned; the second chapter focuses on the communication hardware and firmware between the Digitizer board and the ReadOut Controller (ROC) board, focusing in particular on the communication protocol; in the third chapter this same protocol is modified to achieve the desired performance (200 Mbps); in the fourth chapter an optical fiber communication to Data Acquisition server (DAQ) is realized in collaboration with FCC for fast data acquisition and BER evaluation; in the fifth chapter new communication architecture and protocol are proposed to improve performance without hardware cost.

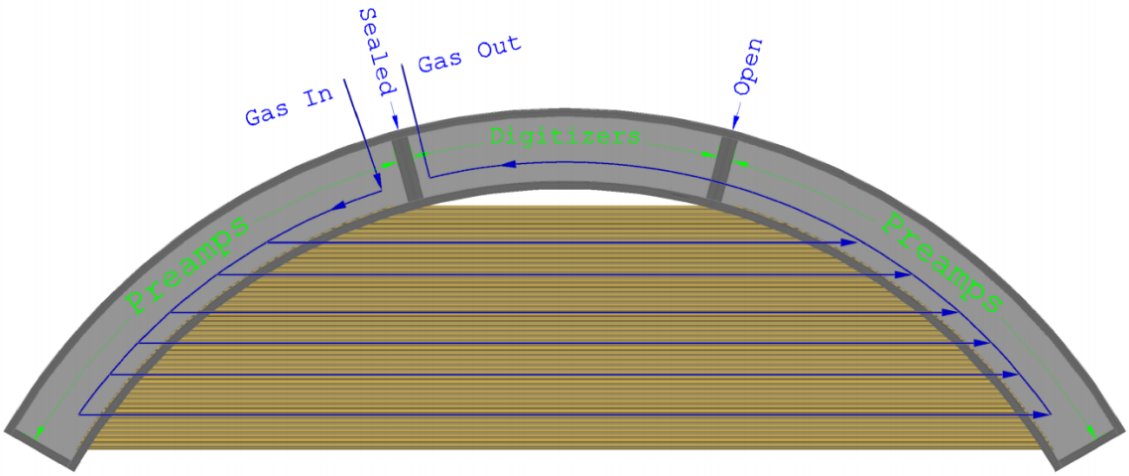
Index

Introduction

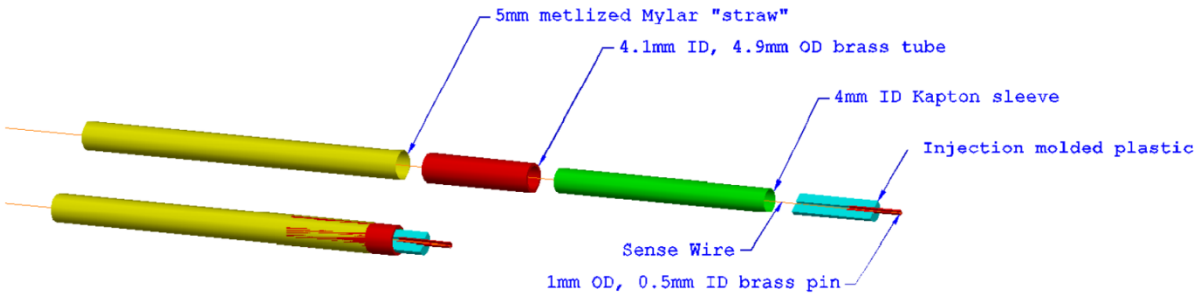
Fermi National Accelerator Laboratory and the Mu2e Collaboration, composed of about 155 scientists and engineers from 28 universities and laboratories around the world, are working to create a new facility to study charged lepton flavor violation using the existing Department of Energy investment in the Fermilab accelerator complex. Mu2e proposes to measure the ratio of the rate of the neutrinoless, coherent conversion of muons into electrons in the field of a nucleus, relative to the rate of ordinary muon capture on the nucleus. The conversion process is an example of charged lepton flavor violation (CLFV), a process that has never been observed experimentally.



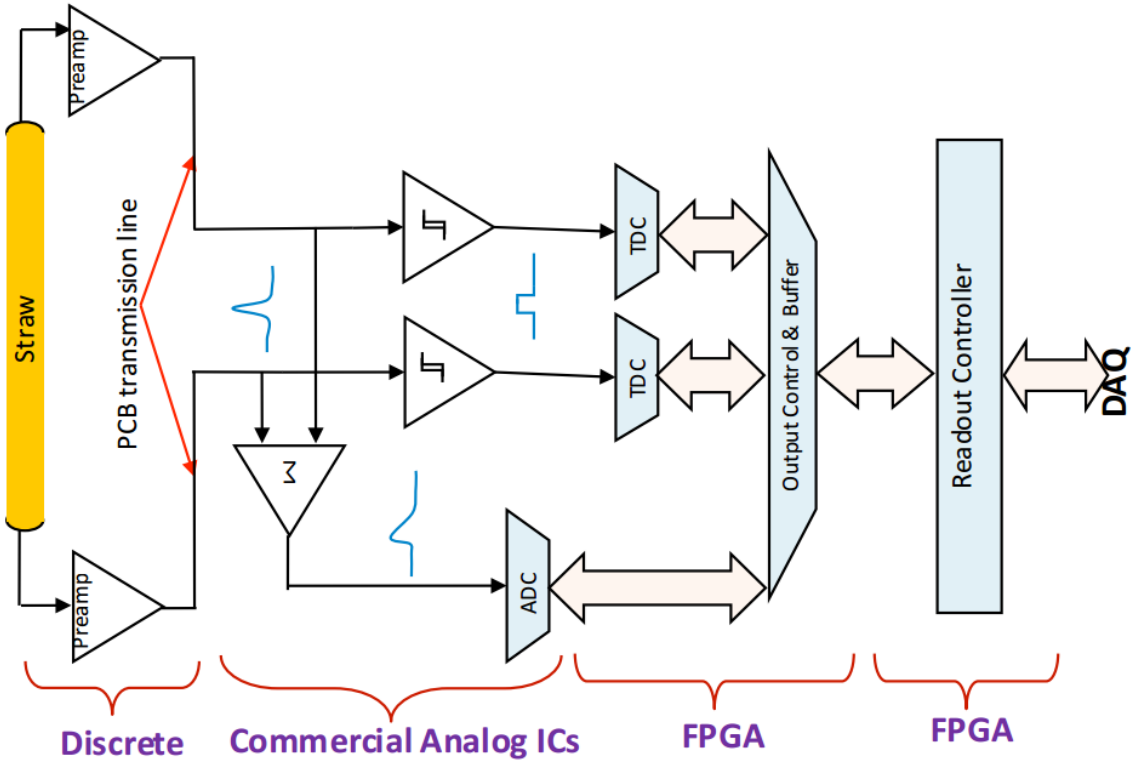
The Mu2e detector shown above consists of a tracker, a calorimeter, a stopping target monitor, a cosmic ray veto, an extinction monitor and the electronics, trigger and data acquisition required to read out, select and store the data. The calorimeter provides independent measurements of energy, position and time, the cosmic ray veto identifies cosmic ray muons traversing the detector region that can cause backgrounds and the extinction monitor detects scattered protons from the production target to monitor the fraction of out-of-time beam.



The tracker, subject of this report, provides the primary momentum measurement for conversion electrons and it must provide this functionality in a relatively unique environment. The tracker resides in the warm bore of a superconducting solenoid providing a uniform magnetic field of 1 Tesla; the bore is evacuated to 10^{-4} Torr. The selected design is a low mass array of straw drift tubes aligned transverse to the axis of the Detector Solenoid, referred to as the T-tracker. The basic detector element is a 25 μm sense wire inside a 5 mm diameter tube made of 15 μm thick metalized Mylar®, referred to as a straw.



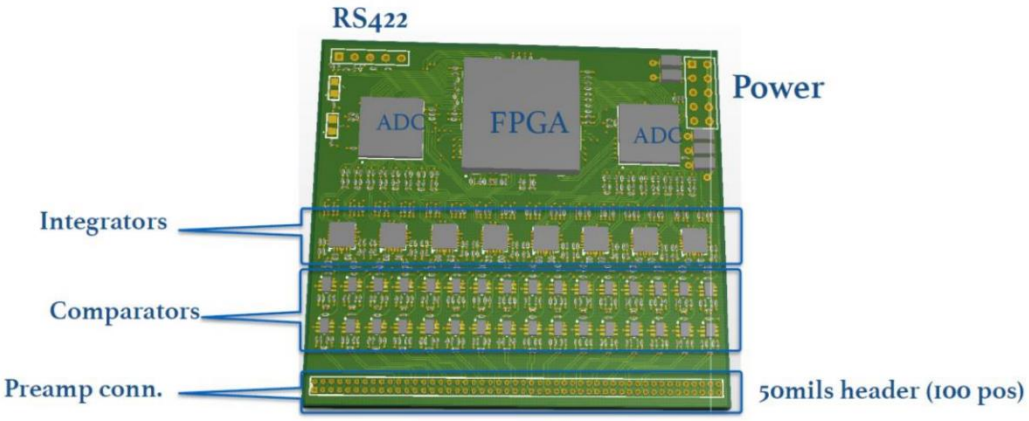
Each straw is instrumented on both sides with preamps and TDCs. Each straw has one ADC for dE/dx capability. To minimize penetrations into the vacuum, digitization is done at the detector with readout via optical fibers. Electronics at the detector will *not* require an external trigger: all data will be transferred out of the vacuum to the DAQ system, and a trigger may be implemented as part of the DAQ.



The tracker uses “time division”: pulse timing is measured at each end of the straw in order to measure the position along the wire. Resolution is ~3cm; this is used for pattern recognition before using stereo for getting a more accurate position. Pulse height (measured through ADC) is measured to provide dE/dx for particle identification.

After amplification and shaping, the analog signal is sent on micro-strip transmission line to the digitizers. Bringing signals from the two sides to a single digitizer board reduces concerns of clock synchronization at the sub-nanosecond level needed for time division. A panel requires six digitizer boards, each servicing 16 straws. Each digitizer board contains: one FPGA with 32 TDC channels; two ADC chips, 8 channels each; and associated analog circuitry. The analog input stage of the digitizer is composed of two comparators to send signals for timing measurement. These work with the preamp

pulses at full bandwidth. In addition, there is a summation and integration amplifier to feed a reduced-bandwidth signal to the ADC.



A completed board is shown in the figure shown above. The ADCs are a commercial 8 channel, 50MSPS, 12 bit devices with high speed serial output. Each ADC digitizes continuously and sends data, without zero-suppression, to the digitizer FPGA. This FPGA also functions as the TDC for the corresponding straw. The ADC clock comes from the digitizer FPGA to ensure the TDC and ADC data remains synchronized. The FPGA combines internal TDC data with external ADC data. ADC zero suppression is achieved by sending only ADC data associated with a TDC hit. The TDC is implemented in the FPGA as a combination of a delay chain for fine timing, and a counter running at 62.5 MHz for coarse timing. Per-stage delay in an FPGA varies with temperature and voltage. An automatic calibration procedure is built into the FPGA to measure and correct for this. The TDC intrinsic resolution is ~25 psec. Folding in comparator jitter, noise, and other external effects the final resolution is ~40 psec. For comparison, the resolution from time division is >80 psec.

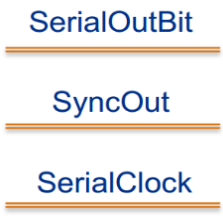
The ROC's primary function is to receive data from the digitizer boards, buffer the data, and then transmit it to the DAQ system. Buffering is needed to continue transferring data during the beam inter-spill time. (We expect to take 8-20 cosmic ray data during the inter-spill time – the front end must remain live – but rates are very low.) The connection from ROC to DAQ is via 2.5 Gbps full-duplex fiber optic links arranged in rings with multiple ROCs per ring, as seen in the picture below.

The ROC includes external DRAM for buffering and links the experiment's Slow Controls system to the digitizers and preamps. DACs, ADCs, and sensors are distributed through each panel and connect to the ROC via SPI and I^2C. To reduce the number Chip Enable lines (SPI normally uses one per chip), SPI port expanders will be used.

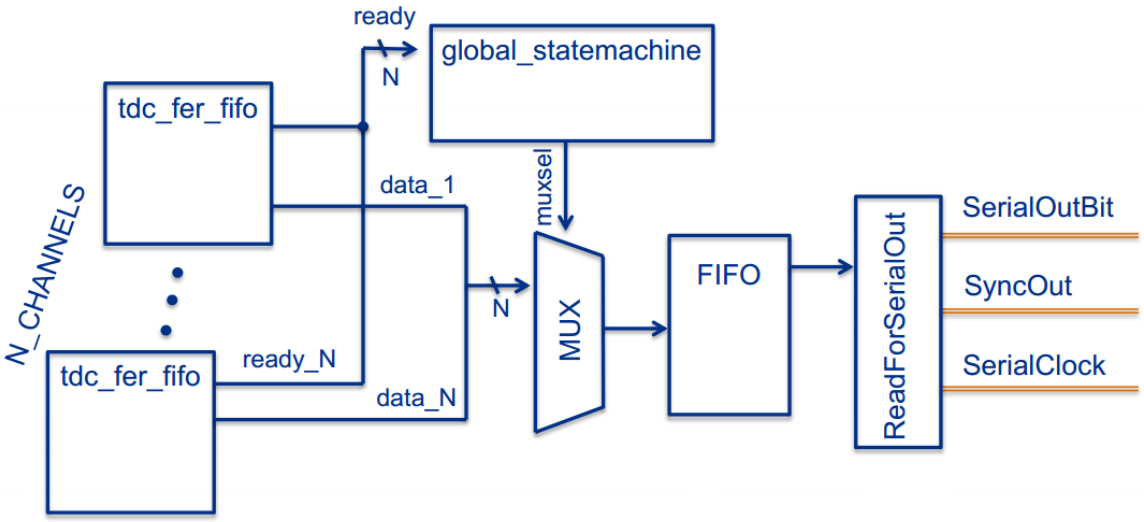
Communication hardware and protocol

Communication between the digitizer and ROC is via LVDS signals. LVDS (Low-Voltage Differential Signaling) is a technical standard that specifies electrical characteristics of a differential, serial communications protocol. LVDS operates at low power and can run at very high speeds using inexpensive twisted-pair copper cables. Since LVDS is a physical layer specification only, many data communication standards and applications are used on top of it. The current implementation of such channels includes three vias used as follows:

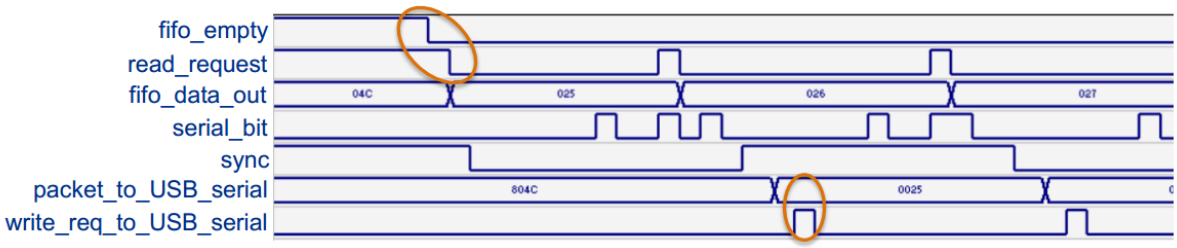
- Information lane (data)
- Packet synchronization lane (frame)
- Clock synchronization lane (clock)



The protocol implemented for this paper includes data flow control from the Digitizer's internal RAM towards the communication channels. The overall block system for the Digitizer is shown in the picture below.

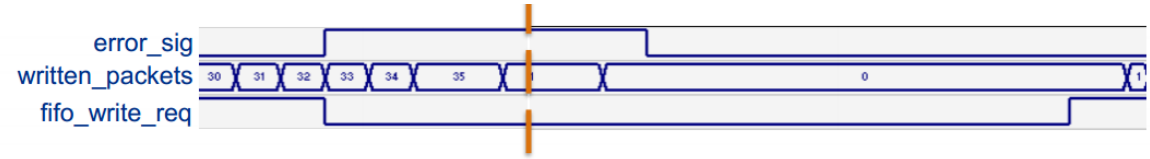


The "ReadForSerialOut" block is assigned to make the serialization of 12-bit-words coming from the internal memory. As soon as the memory block (FIFO: First In First Out) shows that is not empty (just setting the empty signal to zero) the RFSO block start reading out data and serializing it to the communication channels, also managing the SyncOut signal as a frame. This basically implements the control flow as presented in the picture showing the signals below.



From the ROC side there is the “SerialIn” block that de-serializes data coming from the communication channels and elaborates it for storage inside the internal RAM. From the picture on the bottom of the previous page it is clear that data is valid when the write enable signal is set high, i.e. it is the right moment for sampling data in the ROC’s FIFO.

As far as Digitizer’s FIFO overflowing handling is concerned instead, the RFSO block issues an error as soon as it senses the words inside the FIFO going over a preset threshold. This effectively disables the FIFO and the purpose is to keep that memory disabled until the serializer block finishes sending out the last word stored. Then the error signal goes to zero again and the Digitizer starts storing information again in the FIFO made enabled again. The error signal in this stage of communication debugging is used as a write enable. In the picture below there is the elaboration of the signals and the line dividing it means there is some time that is not actively shown in the picture itself.

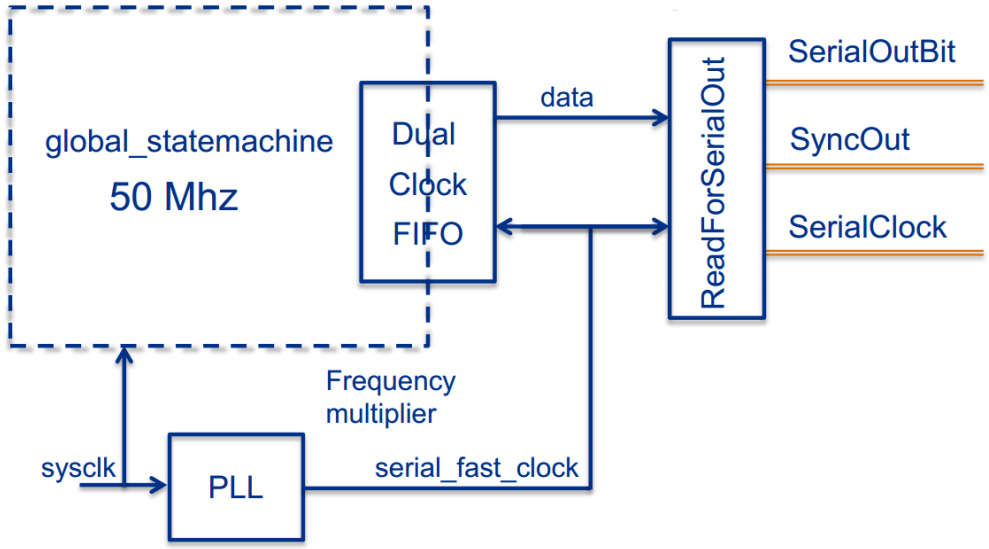


Moreover, this error flag is also sent to the ROC as part of the packet. The packet is then composed of the 12 bit word (sent from the MSbit) and the error flag as the 13th bit.

The protocol implemented here is not to be considered as final. In fact the final design will be composed of four LVDS lines (an additional line for data). However, depending on final FPGA selection, and board layout issues, we are exploring options such as using 8b/10b or similar SERDES (self-synchronizing) data transfer. Since both the digitizer and ROC are FPGA-based, data format is flexible so even the 13-bit packet will be changed completely depending on how the system will be developed. Including rate-leveling, and averaging over a microbunch, the highest rate for any 4-straw group (corresponding to one digitizer data line to the ROC) is 240 kHz or 30 Mbps (at 128 bits/hit). The maximum rate allowed by the LVDS lines is 200 Mbps per data line. There is plenty of head room for adjusting data format, or unexpectedly high rates. A more thorough exploration of communication architecture and protocol considering a given hardware is described in the final part of this report.

Communication speed

The purpose for the communication channel is to transmit data efficiently enough to keep up with the sampling rate. For this reason, the serial communication must be faster than the clock frequency at the writing stage of the internal memory. The internal clock frequency for the overall system (that is also the sampling frequency used on the 12 bit ADC) is 50 MHz and the target for communication speed is 200 Mbps (this is as a first design specification, without architectural adventages or tricks such as 8b/10b or multiple edge transmitting), so four times higher than the system clock speed. In order to do this the communication must have a dedicated clock signal and this is achieved by implementing a PLL with frequency multiplication as shown in the block system below.

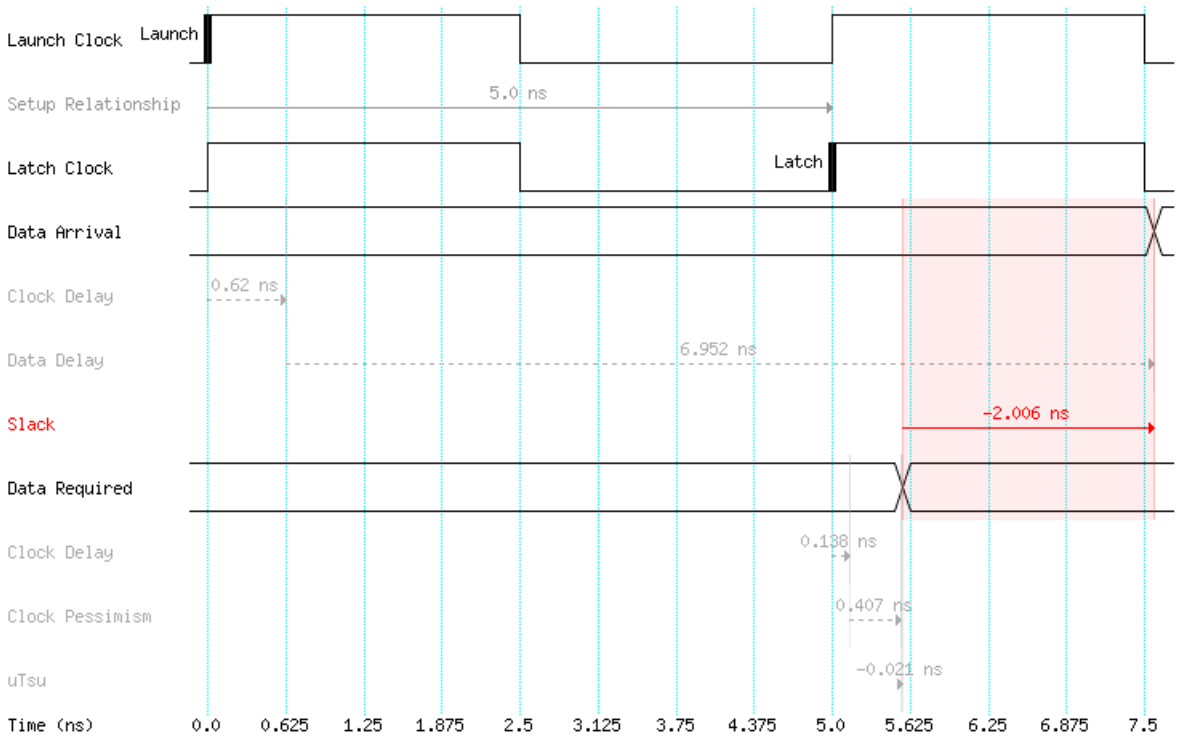


The problem in having this communication faster is that synthesis must be compatible with the target clock frequency. In fact, implementing the system as shown above was not enough. The first message that the compiler outputs is very clear:

```
Critical Warning (332148): Timing requirements not met
```

This message means that the current code cannot be run at the selected frequency and this is even more clear from the Timing Analyses that return the maximum frequency for every synchronization signal. The results of such analyses report that the section regarding transmission cannot run faster than 145 MHz. This is because the delay caused by combinatorial logic between registers is longer than the period of the synchronization signal, i.e. the clock. The delay is caused by intrinsic delay in signal propagation inside electronic components and it is calculated after the Place and Route operation when all lengths of the routing network are known. An example of delay caused by combinatorial logic that exceeds the timing requirements can be found in the scheme on the top of next page. This scheme is generated directly by the Timing Analyzer tool in Altera Quartus software. In the picture it is highlighted that the instant in which data is required occurs roughly 2 ns before it effectively arrives. This forbid the communication system to run at 200 Mbps.

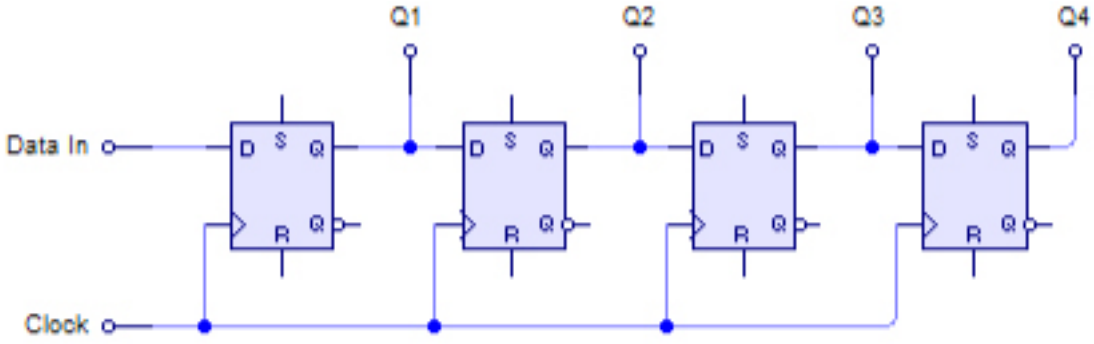
Using the Timing Analyzer tool it is also possible to locate those paths that cause problems as far as delay is concerned. In this case the Worst Path analysis reports that a determined variable called “my_count_curr” of type integer has exactly the worst possible delay. This variable is used in the ReadForSerialOut block as a pointer to the bit that has to be transmitted in the current clock cycle.



It is commonly known that sum operations may cause problems in performance constrained systems because of carry propagation through half adder / full adder architectures. A simple way of solving this problem (still holding the structure of summation) is reducing the width of the counter manually. So it is possible to switch the counter type from integer (32 bits) to std_logic_vector (that has tunable bitwidth). The correct amount of bits to be used in the counter should take into account the maximum value the counter itself reachs before getting reset, i.e. 13 (size of the packet to be transmitted). So the correct calculation is:

$$\lceil \log_2 13 \rceil = 4$$

This solution of course reduces the chance of having problems at high speed, but to achieve maximum performance there has to be a structural change. So, instead of having a sum for pointing the right bit to transmit, a good solution is to have a shift register and transmit always the most significant bit (that is position Q4 in the picture).



At every clock cycle the core operation becomes a simple shift that is not a bottleneck anymore as far as performance is concerned. To verify whether the transmission terminated or not, the shift register is implemented to have an additional bit (in the less significant position) that is set to one in

reset mode. Every clock cycle the ReadForSerialOut block inputs a zero in the shift register and so the termination condition becomes an OR that inputs all the bits of the registers minus the most significant two. This large fan in OR could become a bottleneck in ultra-high performance systems but it does not bother this synthesis and holds correctly 200 MHz clocking.

However, even considering these structural changes in the firmware, the Timing Analyzer tool returns the same “Timing requirements not met” warning. Studying again Worst Path analyses the report shows the following message:

1	-0.856	custommaster:inst3|digi:digi_1|myfifo:outfifo|dcf...o_component|dcfifo_6ek1:auto_generated|rdptr_g[4]

The path showing incorrect behaviour is inside a predefined Altera block, precisely the FIFO memory inside the digitizer. The simple solution to this is resizing the FIFO reducing its depth, in order to have shorter address routing and internal signal management. The result of this is a rescaling from 2^{14} words of 16 bits to 2^{9} words of 16 bits.

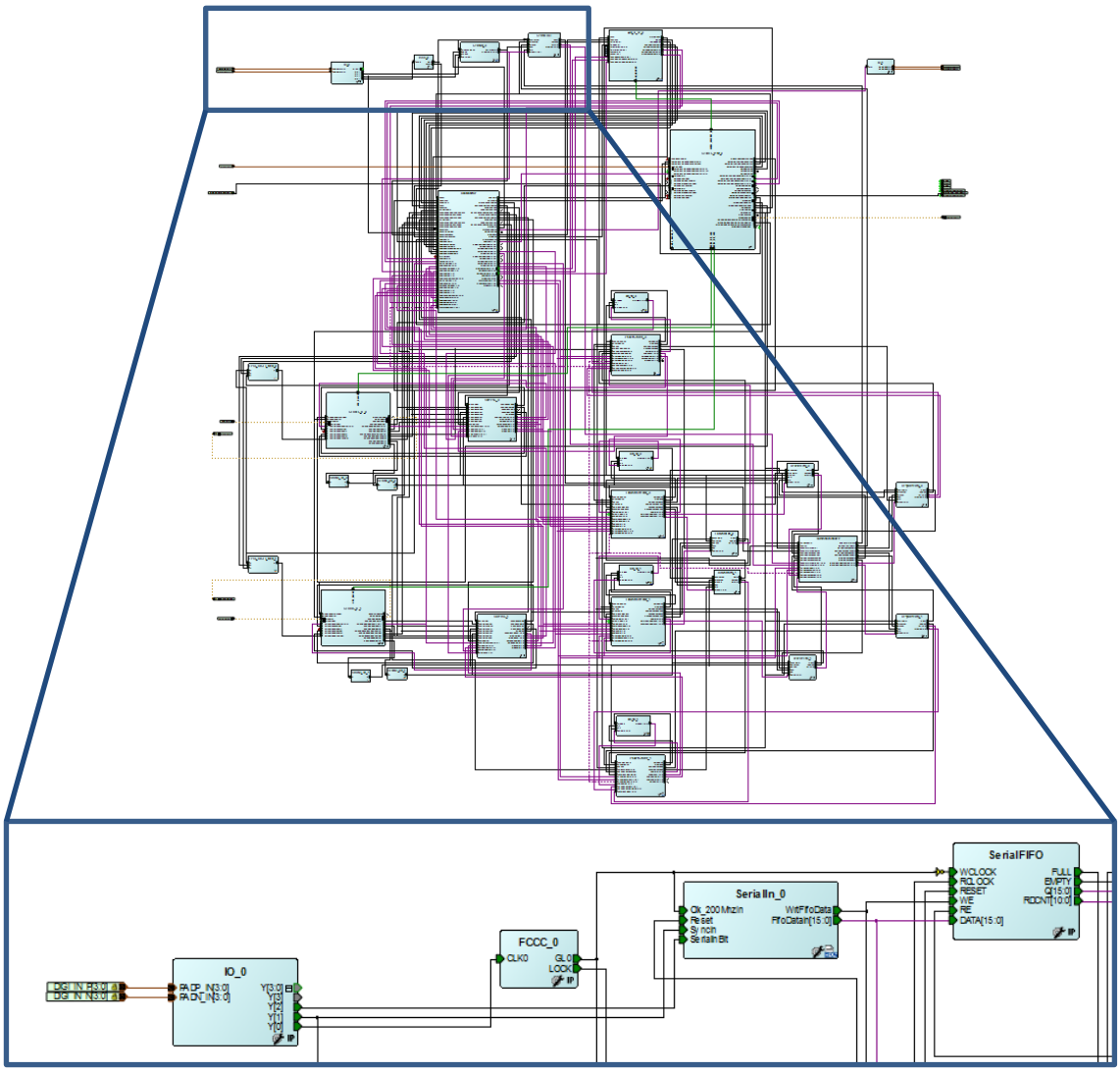
Thanks to all these modifications to the core communication firmware the new result of timing analyses is

	Fmax	Restricted Fmax	Clock Name
2	228.99 MHz	228.99 MHz	inst3|digi_1|b2v_inst1|altpll_component|auto_generated|pll1|clk[0]

that unexpectedly shows also a very good margin (> 10%) from the target set by specifications.

Optical Fiber communication

In order not to have any stall in the ReadOut Controller (ROC) side, the communication to DAQ servers should be done through optical fiber to have the maximum performance possible. This leads to a much more complex system on the ROC but the core section regarding data income from digitizer remains the same.

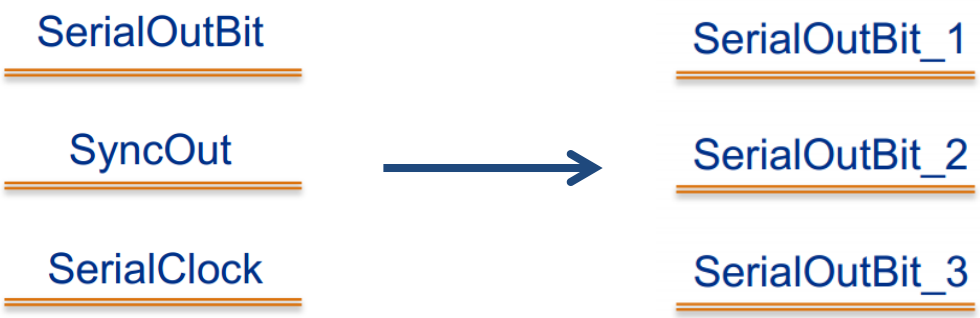


The complexity of the system does not come from the optical fiber itself, which thanks to physical characteristics (excellent BER even at very high speeds) does not need to implement complex communication protocols, but from the interface of the DAQ that is based on the PCIe standard. Moreover the system has also to implement firmware for testing directly on the board if operations are performed correctly. This means Serial Virtualization via USB cable is still present but has a completely different purpose: the aim of this communication is to make internal flags and signals observable for debugging. Some easy examples that can help understanding this functionality are the “write enable” counter, which counts how many times a word is written inside the internal memory of the ROC; the “lock” signal that asserts if the ROC internal PLL is synchronized with the Clock coming through LVDS input (which rules bit synchronization) and the “reset” signal.

Thanks to fast communication toward a DAQ server it is also possible to gather as much incoming words as desired and this is vital for having Bit Error Rate analyses.

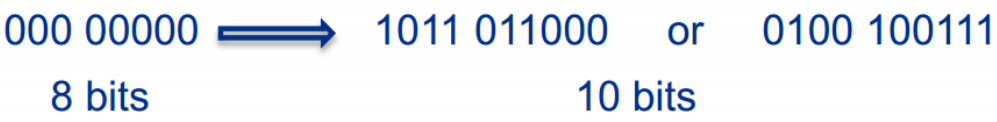
Architecture and protocol modification

Considering that 200 Mbps is not enough for keeping up completely with the sampling rate (because of many ADC and TDC channels at the Digitizer side) it is important to improve the LVDS channels efficiency for speeding up the communication as much as possible. This can be done by changing the communication architecture and protocol. In fact it is immediate to see that if all three LVDS lanes were dedicated to information transmission the overall system would be much faster.



Clearly this transformation is not without complexity and computational cost. In fact, clock synchronization and packet synchronization must be transmitted together with information in order to have a coherent communication.

In order to have clock synchronization together with information it is possible the 8b/10b encoding. This particular coding technique provide enough state changes to allow reasonable clock recovery. This is done by simply implementing a PLL on the receiver side, which will provide exact synchronization. The encoding changes the packet as it is visible in the picture below:



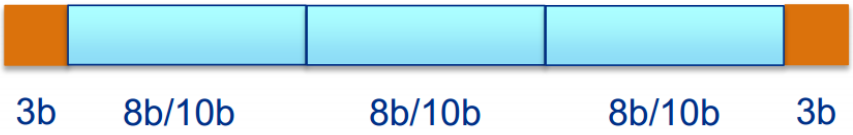
This particular coding also provides DC balance for the signal, which is extremely important for differential amplifiers at the output stage. In fact, in case of DC imbalance there is a bias on the operating point of the amplifier and this could result in additional distortion to the signal. Another additional advantage that such coding carries is that considering that 10-bits words use much less of their available combinations to represent 8-bits words, it is possible to have reserved words for commands or flags, such as the overflow flag explained before.

Moreover, to achieve packet synchronization inclusion, head and tail pattern bits are included. This is necessary to understand where to find the most and the least significant bits written in a word at the receiver. This results in a structure of the packet that looks like the following:

Such implementation has also the advantage of providing even more commutations on the information lane, which improves clock recovery.

To give a realistic example of how this protocol implementation would better the performance of the system, a brief example is presented here. If we assume that s is the maximum speed achievable with the architecture that implements a separate lane for packet and clock synchronization, that the packet is made of 2 words of 12 bits each encoded through 8b/10b, that head and tail patterns are composed of a sequence of 3 bits, the result will be the following:

$$S_{final} = (3 \text{ lanes}) \cdot \frac{24 \text{ information bits}}{36 \text{ bits}} \cdot s = 2s$$



In this particular case then, without any cost on adding hardware to the system but just improving communication efficiency, it would be possible to achieve 400Mbps.

Conclusions

A Serial Communication has been implemented successfully for Mu2e Tracker Electronics. In particular, the communication between the Digitizer board and the ReadOut Controller board is 200 Mbps fast. A successive modification in efficiency could result in twice the speed and is worth being explored thoroughly. Finally, first experimentations on computational architecture and communication through optical fiber toward Data AcQuisition system have been successfully accomplished.