Fermilab

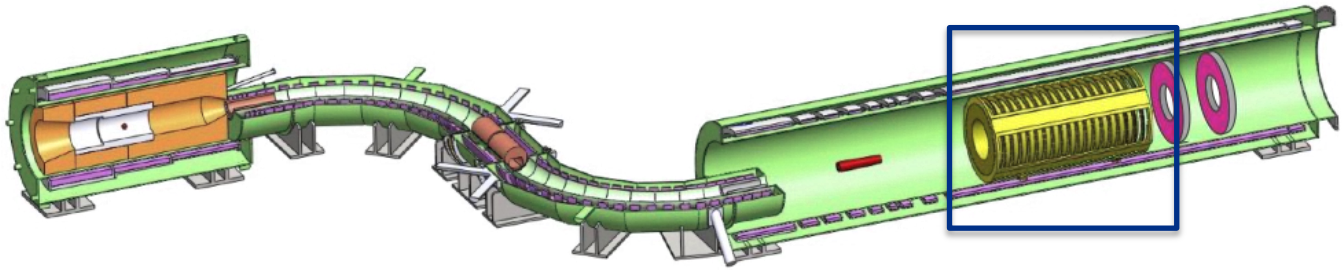
Managed by Fermi Research Alliance, LLC for the U.S. Department of Energy Office of Science

Serial Communication for Mu2e Tracker Electronics

Tommaso Vincenzi
University of Ferrara

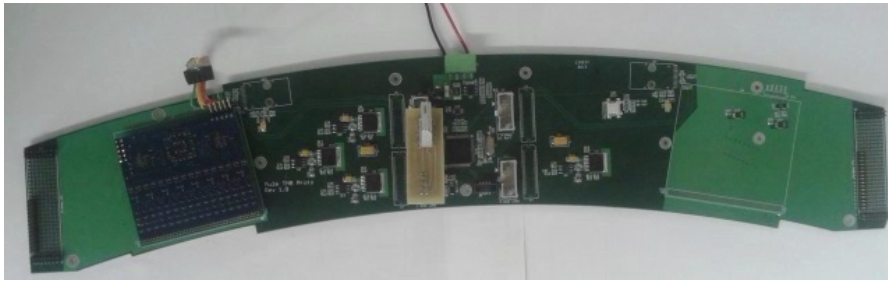
Endterm Presentation
23 September 2015

Introduction



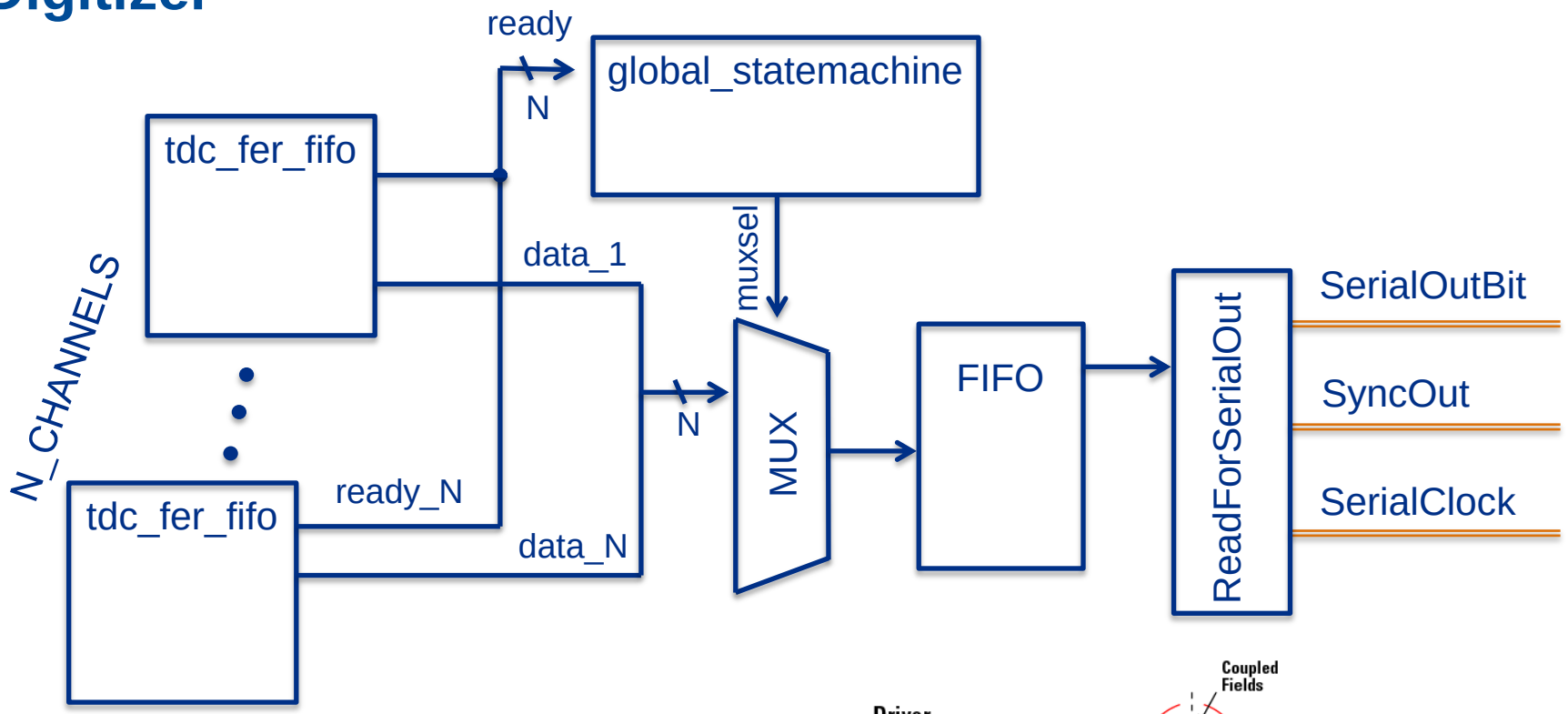
Front-End Electronics

- Digitizer :
  direct acquisition, FPGA (Altera Cyclone III)
  acquires data from ADCs and TDCs



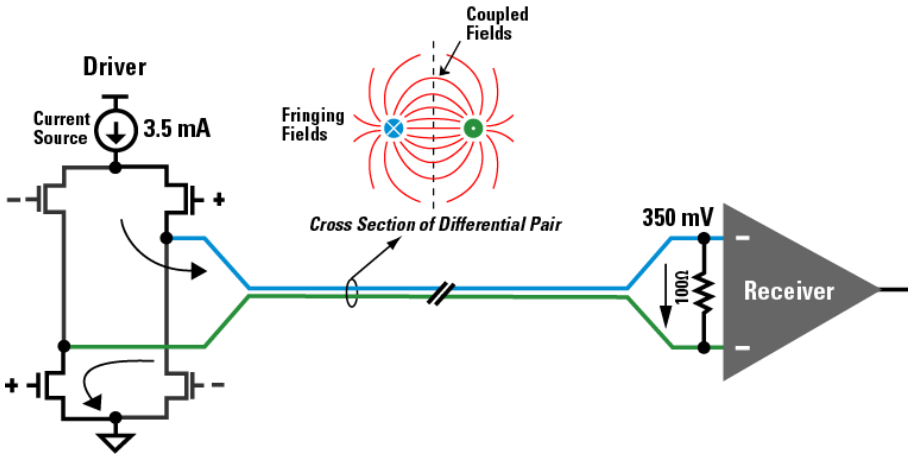
- ReadOut Controller :
  collection, FPGA (Microsemi SmartFusion2)
  handles data flow from the digitizers (6 boards) to the next stages towards the data center (DAQ).

Digitizer



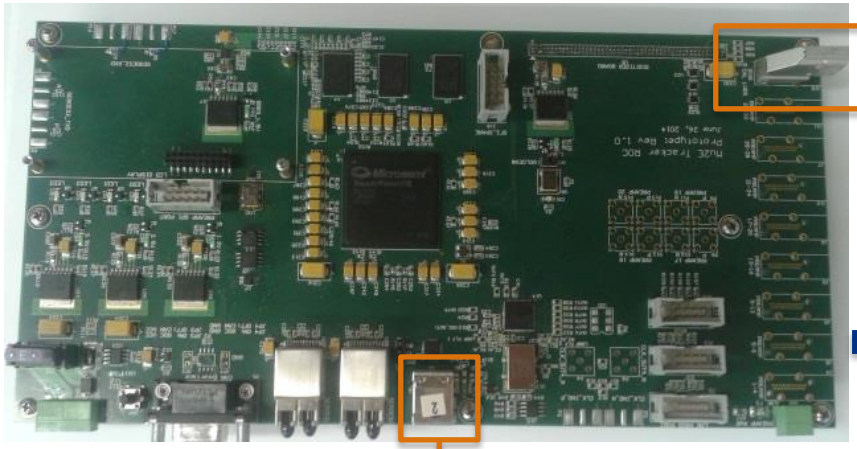
LVDS Out Channels

- Communication standard,
  high speed and low power
- 50 MHz clock frequency



Fermilab

ReadOut Controller



HDMI cable

LVDS input channels

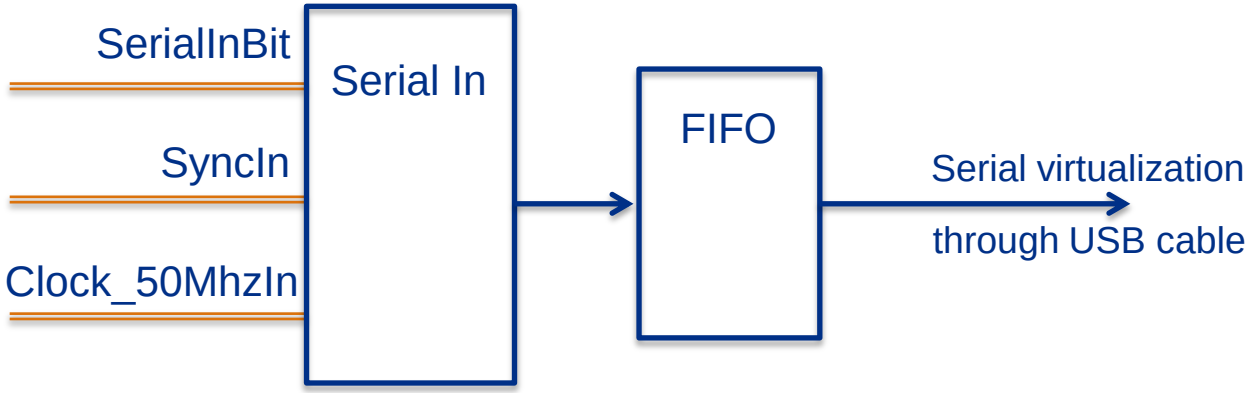
Serial to PC communication

Problem:

USB-Serial speed << LVDS speed (50 Mhz for now)
(~kHz)

⇨ How to test this?

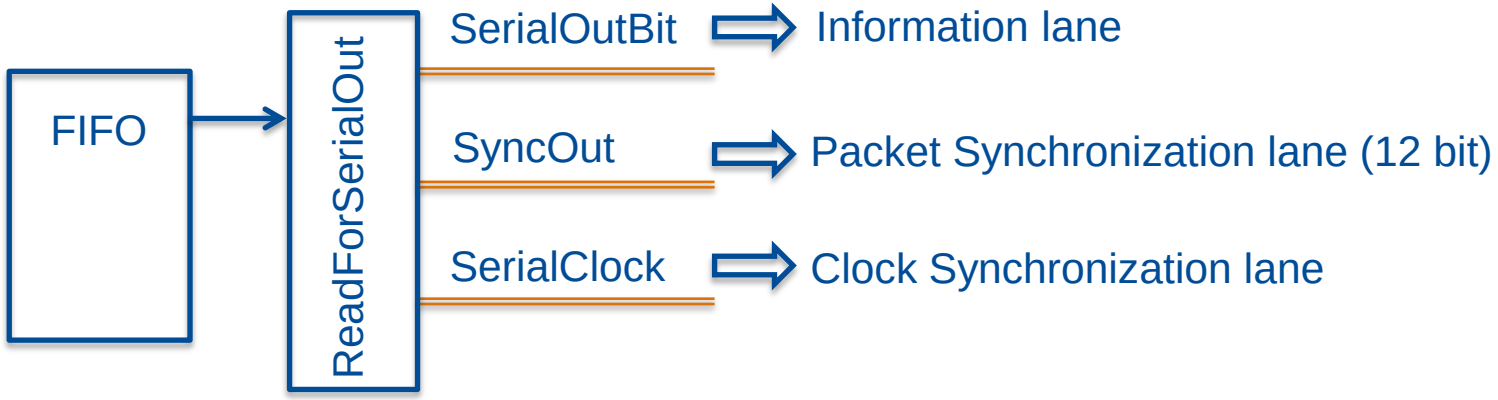
The Digitizer sends data **patterns** (incremental) and the ReadOut Controller stops reading the inputs when the memory is full.
Then it flushes it to the PC to allow **verification**.



⇨ It's also important to have the USB communication as fast as possible in order to acquire large amount of data for **Bit Error Rate** analysis in reasonable time (**920 kHz Standard Max**)

Fermilab

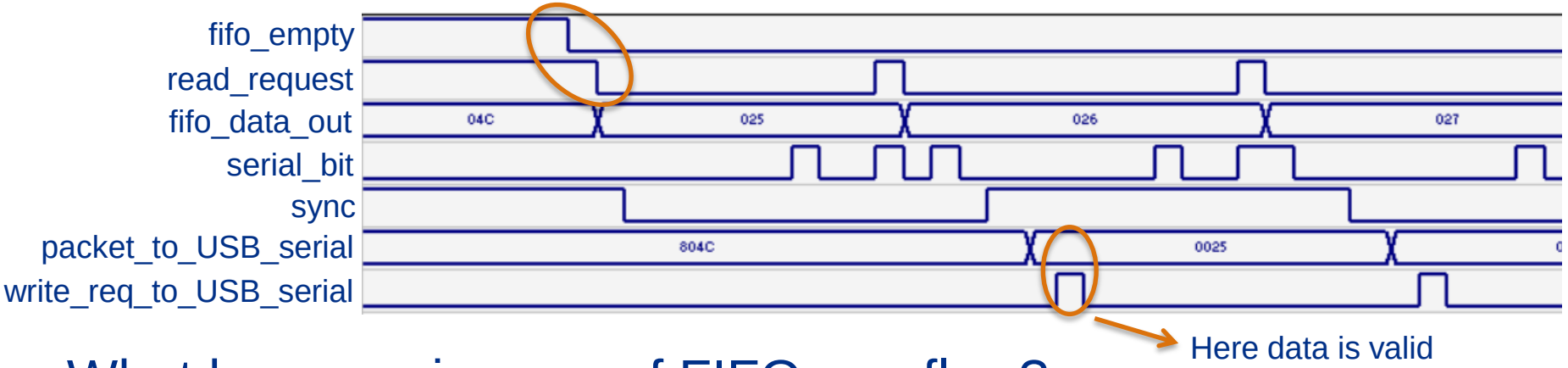
Communication – Step 1



- Flow control needed for streaming data out of the FIFO
- What happens in case of FIFO overflow?
- How fast can the single lane go (Mbit/s)?
- Is this structure resource efficient?

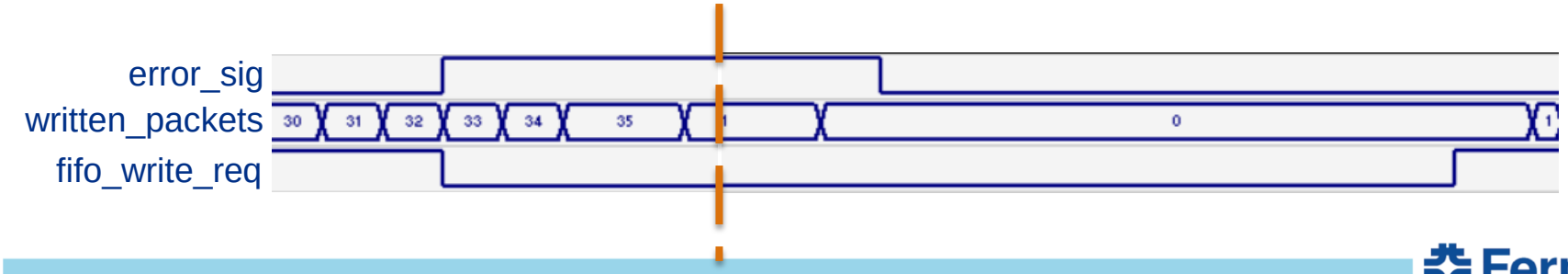
- Flow control needed for streaming data out of the FIFO

  ⇨ As soon as the FIFO block sends a NOTEMPTY signal, the ReadForSerialOut block starts reading the packets out of it and setting the communication towards the ReadOut Controller (ROC)



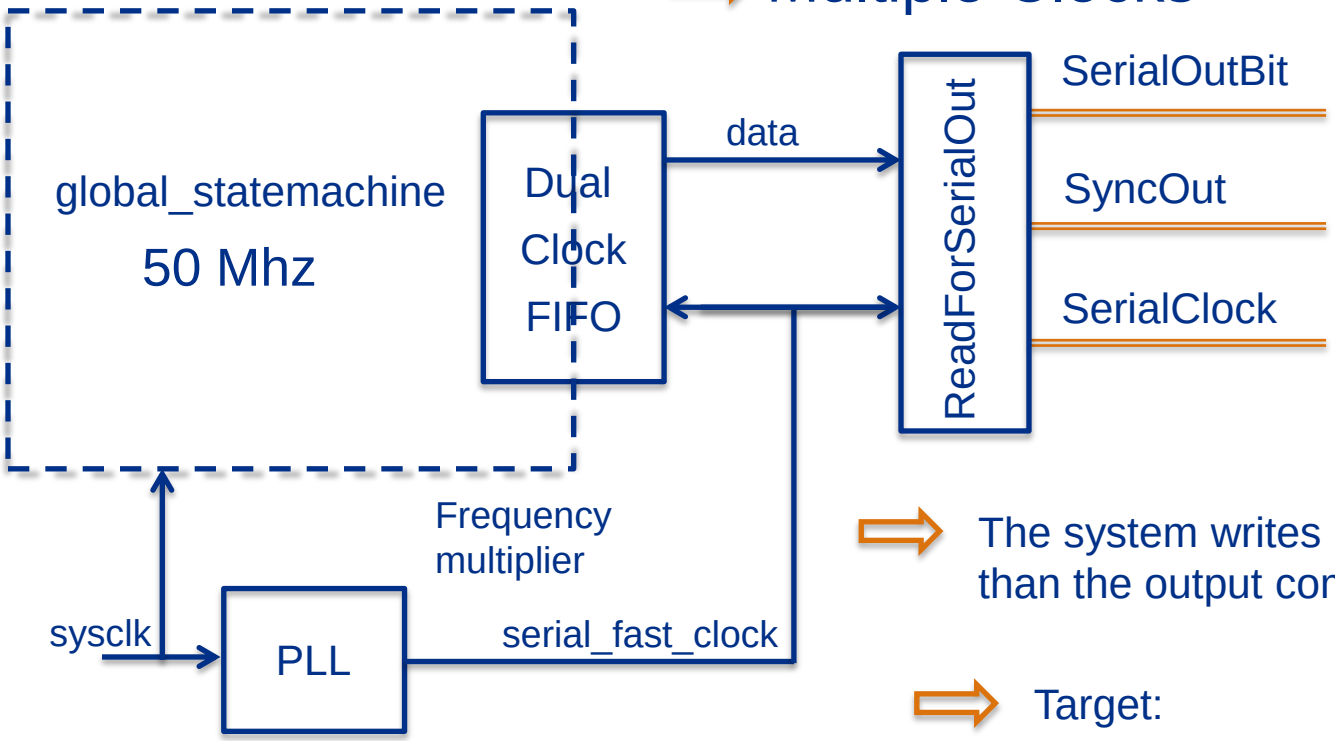
- What happens in case of FIFO overflow?

  ⇨ The packet is composed of 13 bits instead of 12 and the 13th is the **error flag**. As soon as the word number in the FIFO goes **over** a THRESHOLD, the error flag goes up and the FIFO is **disabled**. It stays disabled (and the error flag is kept high) until the FIFO is empty and every packet written before the error triggering is streamed out towards the ROC.



Communication – Step 2

⇨ Multiple Clocks



⇨ The system writes in the FIFO **slower** than the output communication frequency.

⇨ Target:

200 Mbit/s

not working!

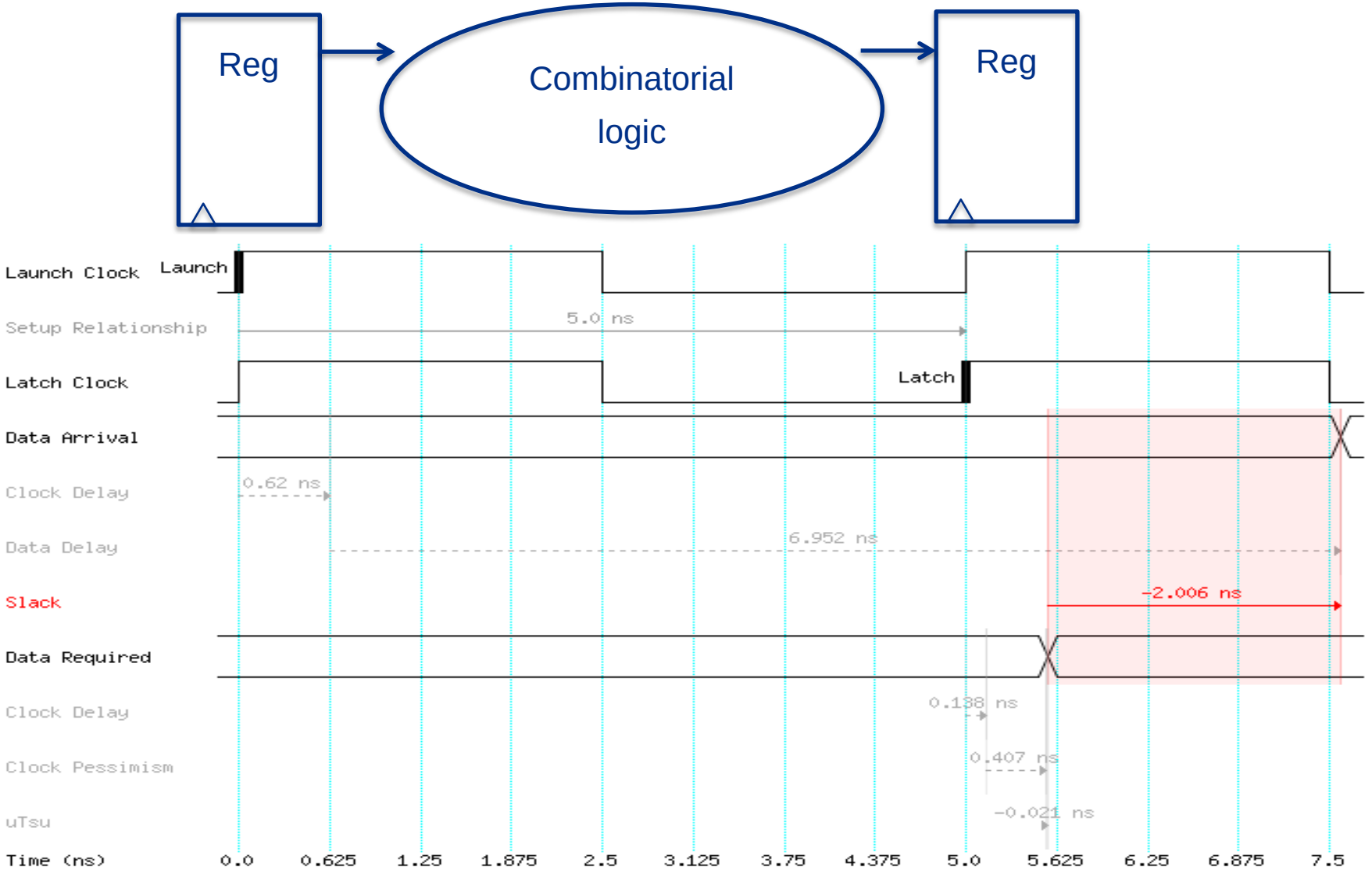
⇨ Testing how **fast** the **single communication lane** can go

Critical Warning (332148): Timing requirements not met

	Fmax	Restricted Fmax	Clock Name	Note
1	122.73 MHz	122.73 MHz	clk	
2	142.73 MHz	142.73 MHz	inst3|digi_1|pll_for_communication|altpll_component|auto_generated|pll1|clk[0]	

Fermilab

Communication – Step 2



Fermilab

Communication – Step 2

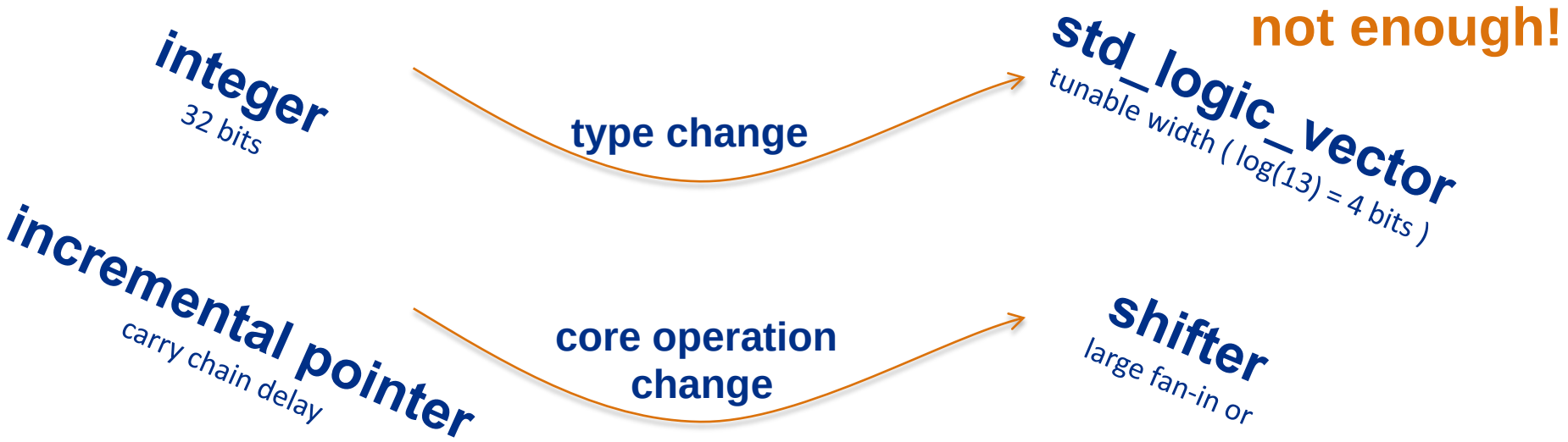
- New firmware structure to have intelligent synthesis
  - **Worst path** analysis to understand where to change it

-2.006 | custommaster:inst3|digi:digi_1|ReadForSerialOut:fifo_handler|my_count_curr[20]

Communication – Step 2

- New firmware structure to have intelligent synthesis
  - **Worst path** analysis to understand where to change it

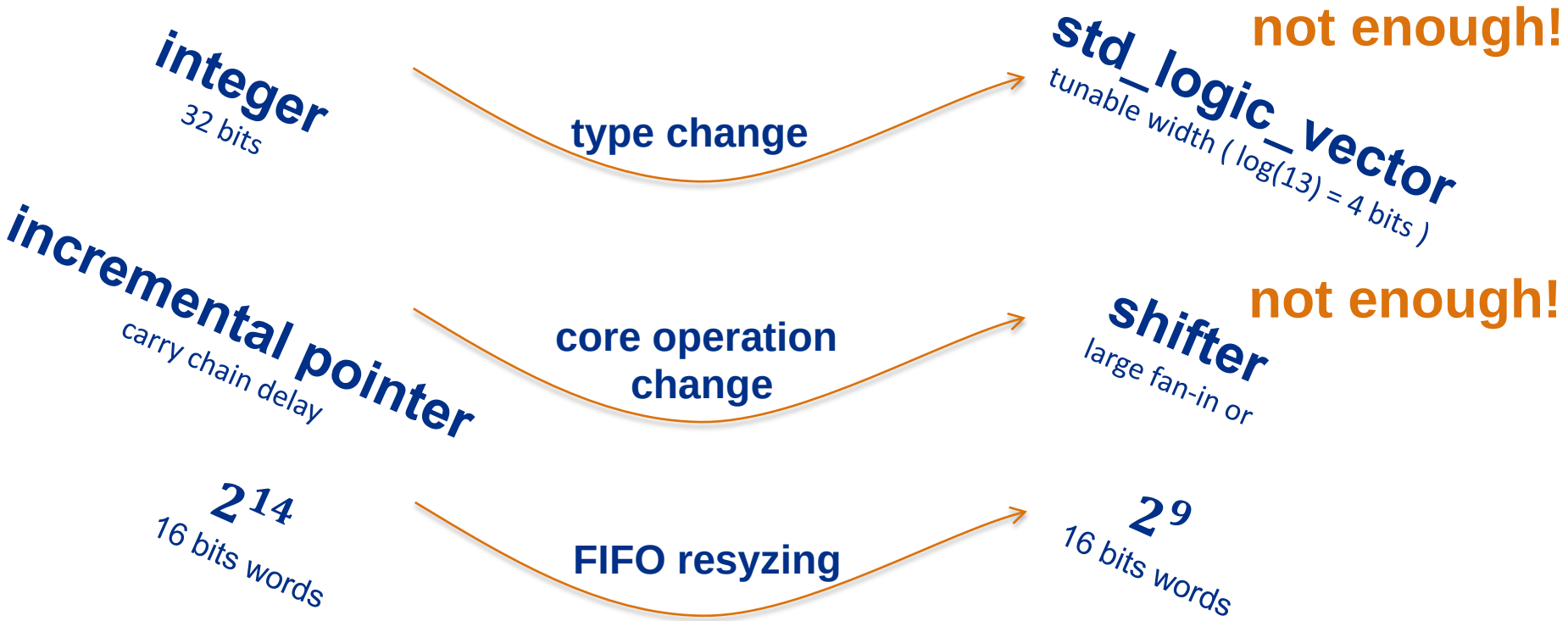
-2.006 | custommaster:inst3|digi:digi_1|ReadForSerialOut:fifo_handler|my_count_curr[20]



Communication – Step 2

- New firmware structure to have intelligent synthesis
  - **Worst path** analysis to understand where to change it

-2.006 | custommaster:inst3|digi:digi_1|ReadForSerialOut:fifo_handler|my_count_curr[20]

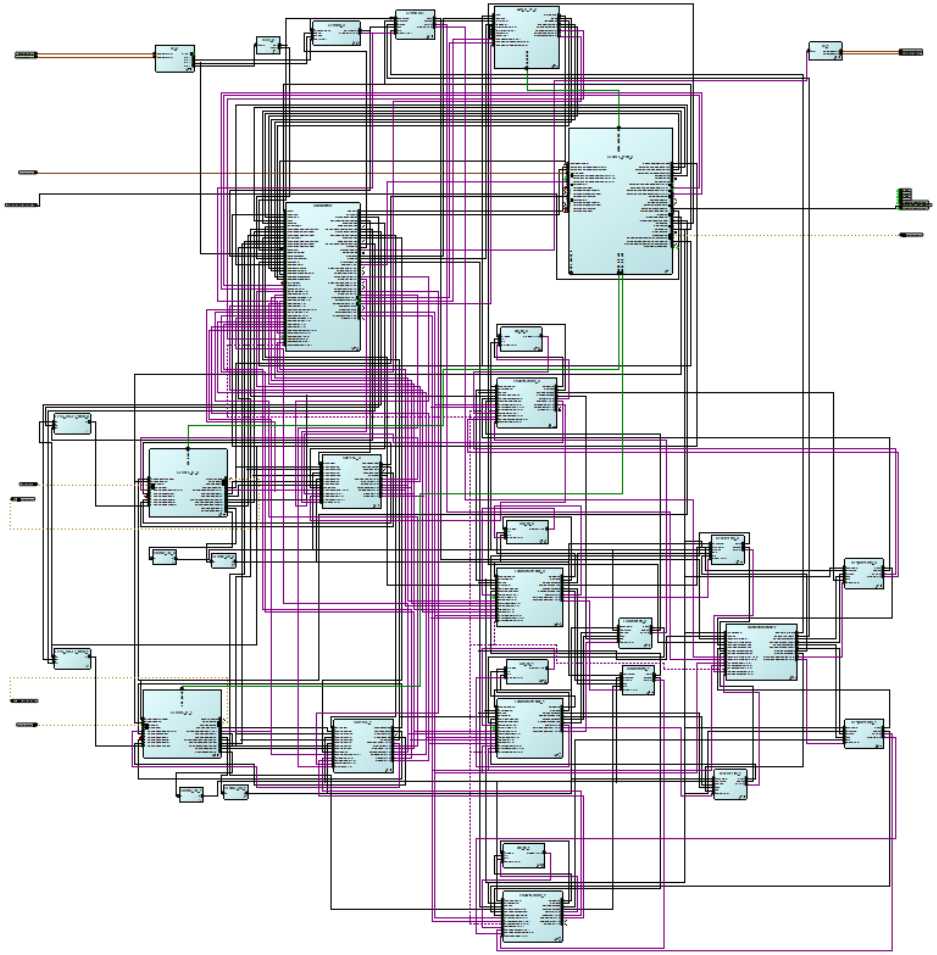


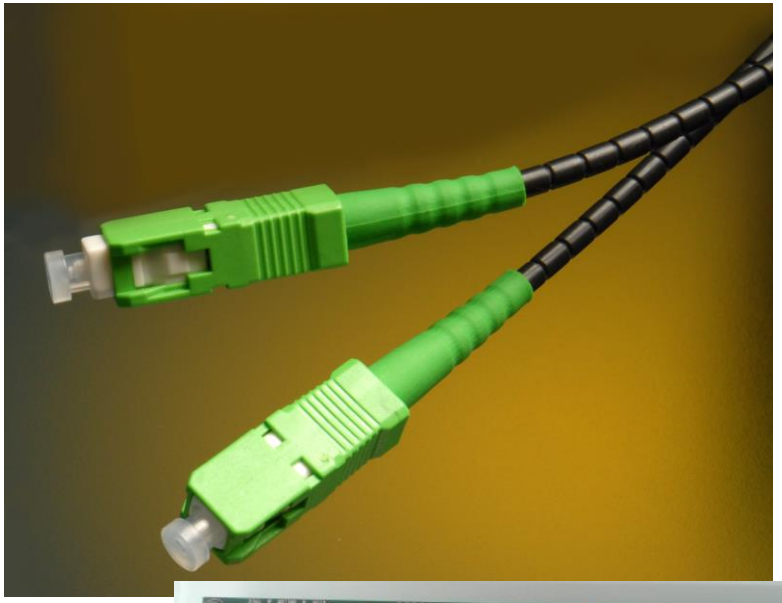
	Fmax	Restricted Fmax	Clock Name
1	142.39 MHz	142.39 MHz	clk
2	228.99 MHz	228.99 MHz	inst3|digi_1|b2v_inst1|altpll_component|auto_generated|pll1|clk[0]

Communication – Step 3

⇨ Optical fiber to PCIe board

⇨ Collaboration with FCC



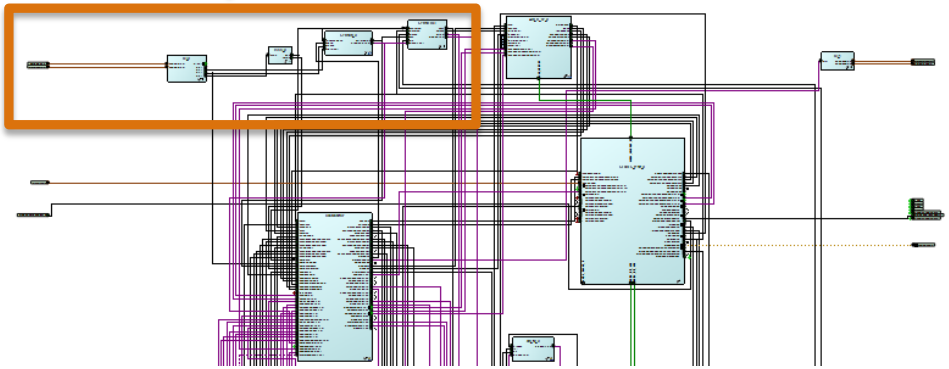


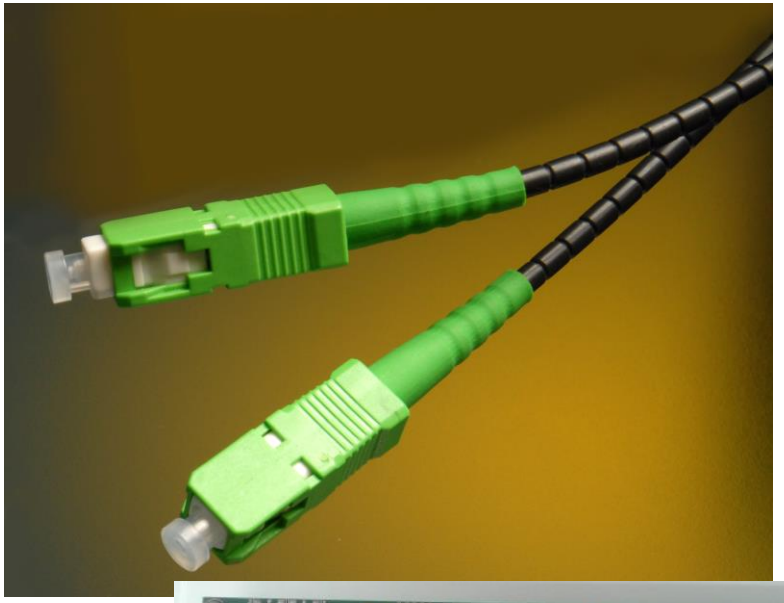
Fermilab

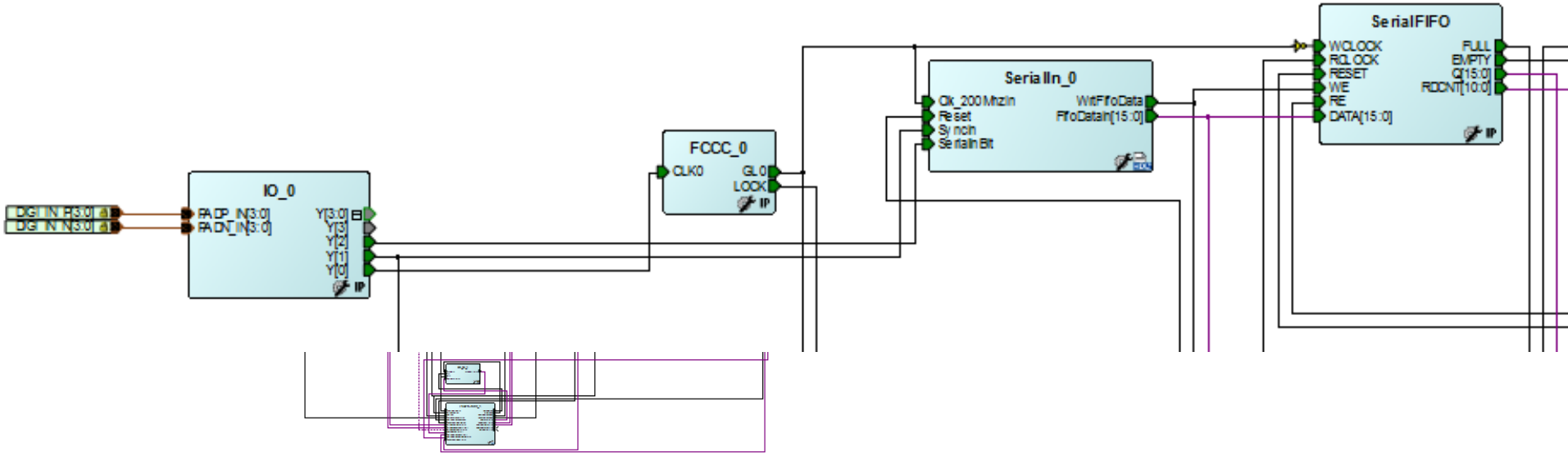
Communication – Step 3

⇨ Optical fiber to PCIe board

⇨ Collaboration with FCC







Fermilab

Communication – Future Steps

Is this structure resource efficient?

SerialOutBit

SyncOut

SerialClock

Communication – Future Steps

Is this structure resource efficient?

SerialOutBit

SyncOut

SerialClock

- 8b/10b encoding:
  data encoding that includes **clock** information

Example:

000 00000 ⟹ 1011 011000 or 0100 100111

8 bits 10 bits

Advantages:

- additional unused words (reserved words for commands, e.g. error)
- DC free signals (very important for differential amplifiers)

Communication – Future Steps

Is this structure resource efficient?

- 8b/10b encoding

SerialOutBit

SyncOut

~~SerialClock~~

- Packet head and tail bits

  packet **synchronization** is obtained inserting determined patterns that are recognized by the receiver

Example:

Fermilab

Communication – Future Steps

Is this structure resource efficient?

- 8b/10b encoding
- Packet head and tail bits

- All lanes are reserved for information transmission

  resources saved thanks to intelligent coding are now available for **information** transmission

Communication – Future Steps

SerialOutBit_1

SerialOutBit_2

SerialOutBit_3

Is this structure resource efficient?

- 8b/10b encoding
- Packet head and tail bits
- All lanes for information

PROBLEM:

timing recovery on ROC side

Example

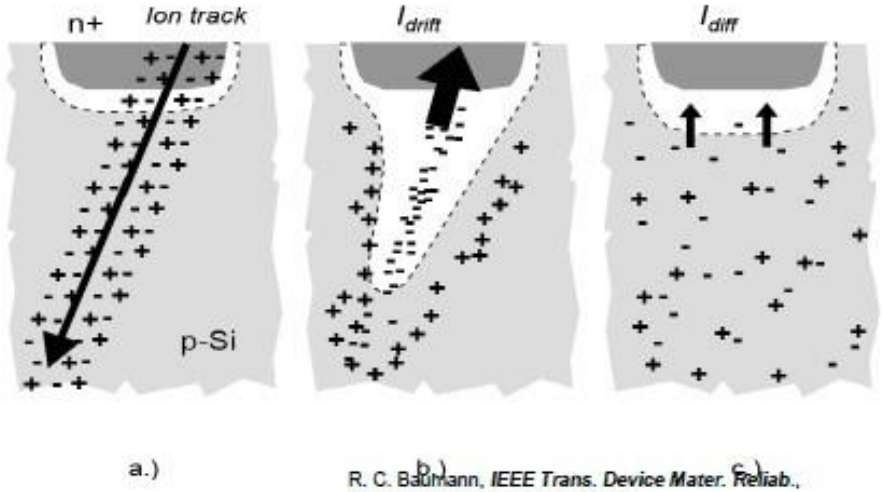
- If **s** is the maximum speed of the single lane
- Words of 12 bits
- Sending synchronization bits every two words

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

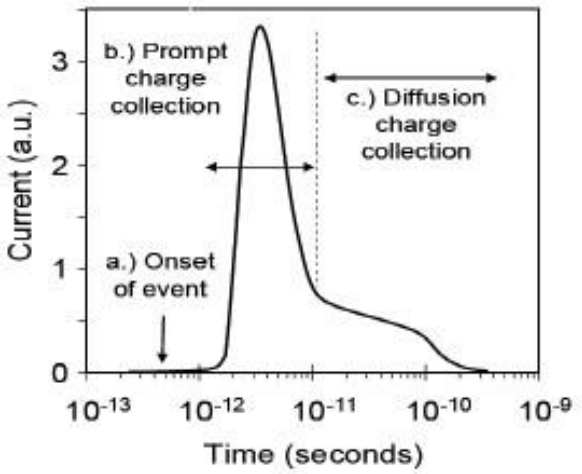
Fermilab

Radiation tests

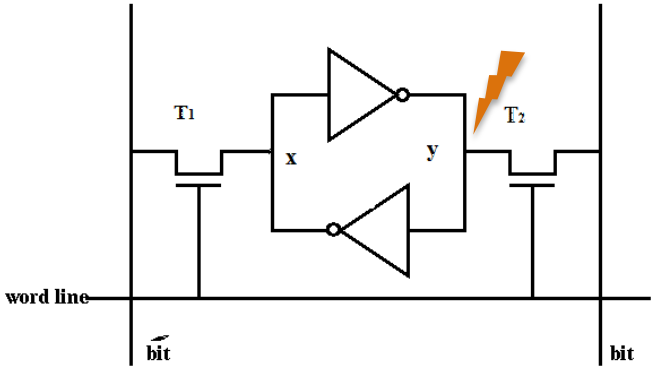
⇨ Soft error due to radiation-induced **particle ionization**



R. C. Baumann, *IEEE Trans. Device Mater. Reliab.*, vol. 5(3), p. 305-316, Sept. 2005



⇨ Serious problem in memories (especially SRAM)

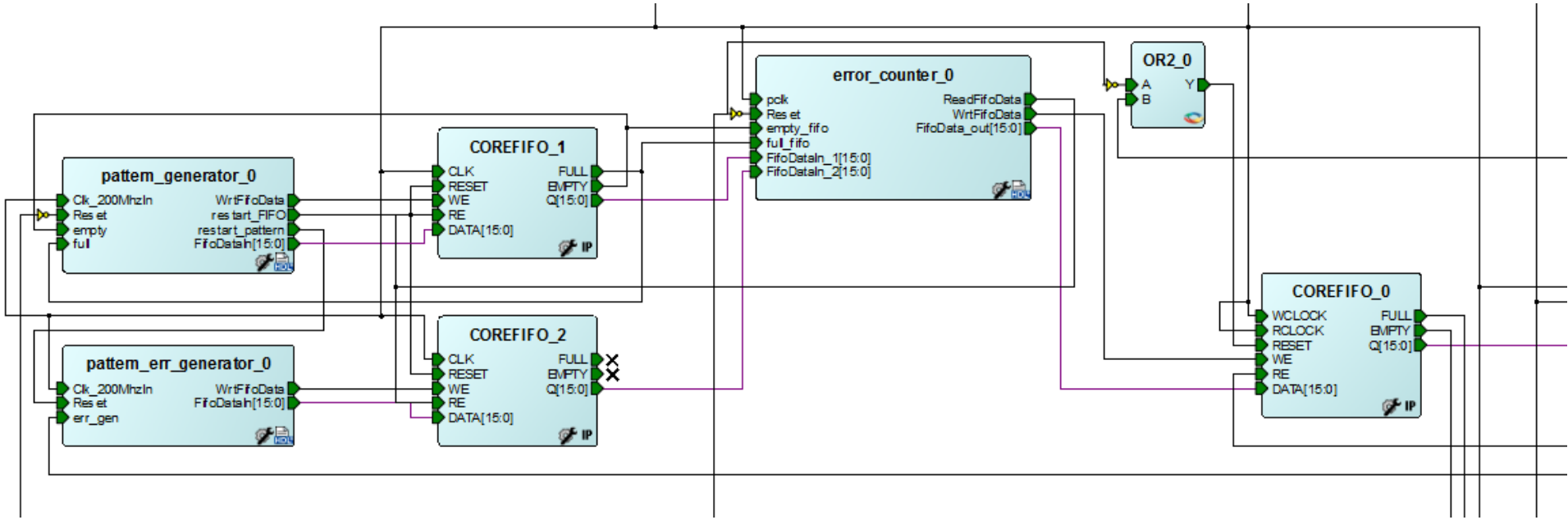


A track in the point shown in the picture could cause a **change of state** and so a permanent error.

Fermilab

Radiation tests

⇨ Adequate firmware structure to test the code



Extensive memory usage in the code:

- 92.75% 1st type : 1K18
- 88.89% 2nd type : 64x18


THANK
YOU