DAPHNE Self-trigger integration

D. Avila, **E. Cristaldo**, I. López de Rego

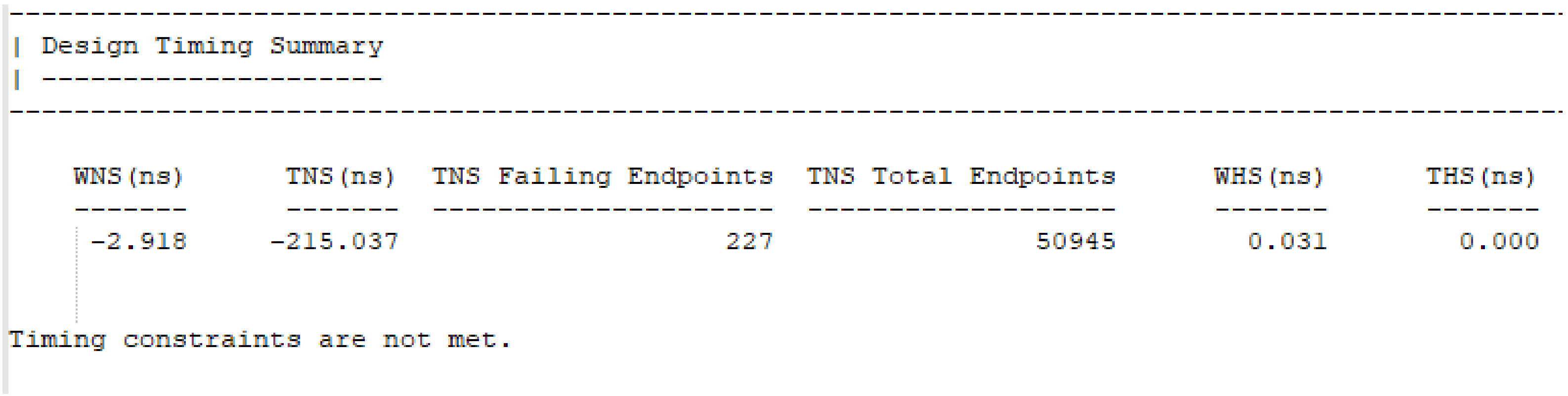
August 15th , 2024

UNIVERSIDAD EIA Ser, Saber y Servir
UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA
INFN Istituto Nazionale di Fisica Nucleare
DUNE

Content

- Timing issues:
  - How debug registers are read in the most updated commit in main.
  - Pipelining: Solution to timing issue.
- Self-trigger integration:
  - Module scheme.
  - New timing issues and solution.
  - Resources utilization.
- Self-trigger simulations:
  - Signals.
  - Trigger jitter.
- Conclusions.

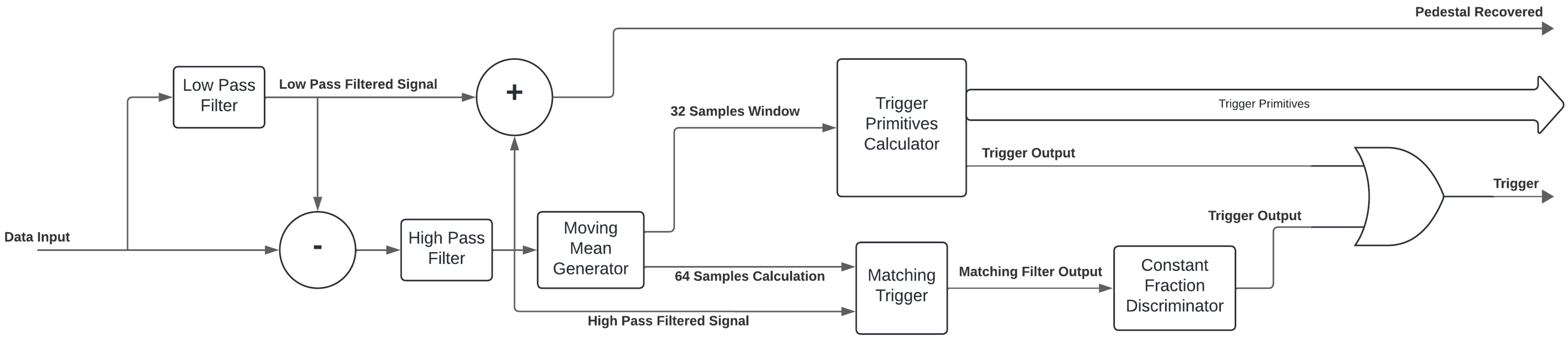
Timing Issues



```
------------------------------------------------------------------------------------------------
| Design Timing Summary
| ---------------------
------------------------------------------------------------------------------------------------

    WNS(ns)      TNS(ns)  TNS Failing Endpoints  TNS Total Endpoints      WHS(ns)      THS(ns)
    -------      -------  ---------------------  -------------------      -------      -------
     -2.918     -215.037                    227                50945        0.031        0.000


Timing constraints are not met.
```

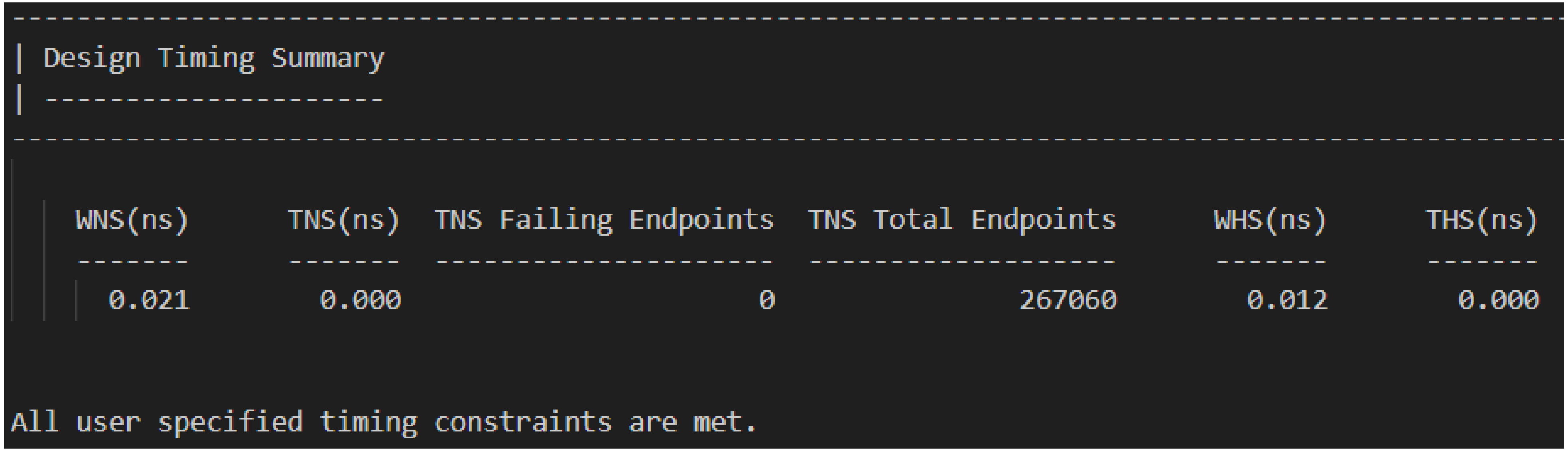
- The latest commit [97823e7] does not met timing constraints. This issue is caused by the introduction of debug registers in the st40_top module, causing congestion due to the large amount of signals wire to the BIG mux in the ethernet data sender.
- Although the mentioned issue, the design was found to be stable during runs because the negative slack does not affect the core data senders and the integrity of data.
- When the self-trigger modules are included, the issue gets dramatically worse to the point where data integrity cannot be guaranteed.
- The issue was solved by pipelining and multiplexing the readout of these registers, but compatibility with the DAQ readout is lost. This solution was not merged yet and exist in a different branch.

Module Scheme



- The trigger aims to be a combination of all three strategies. Since the last results showed that the three modules have similar behaviours, we have decided to use ciemat's strategy in a parallel block to obtain the trigger primitives, avoiding problems with data synchronization since the trigger happens later using eia's matching trigger.
- However, we must keep in mind that by doing this, there is a possibility that sometimes triggers might happen for the matching trigger and not for the ciemat's trigger, therefore creating packet frames with no trigger primitives information, this must be addressed in following weeks.

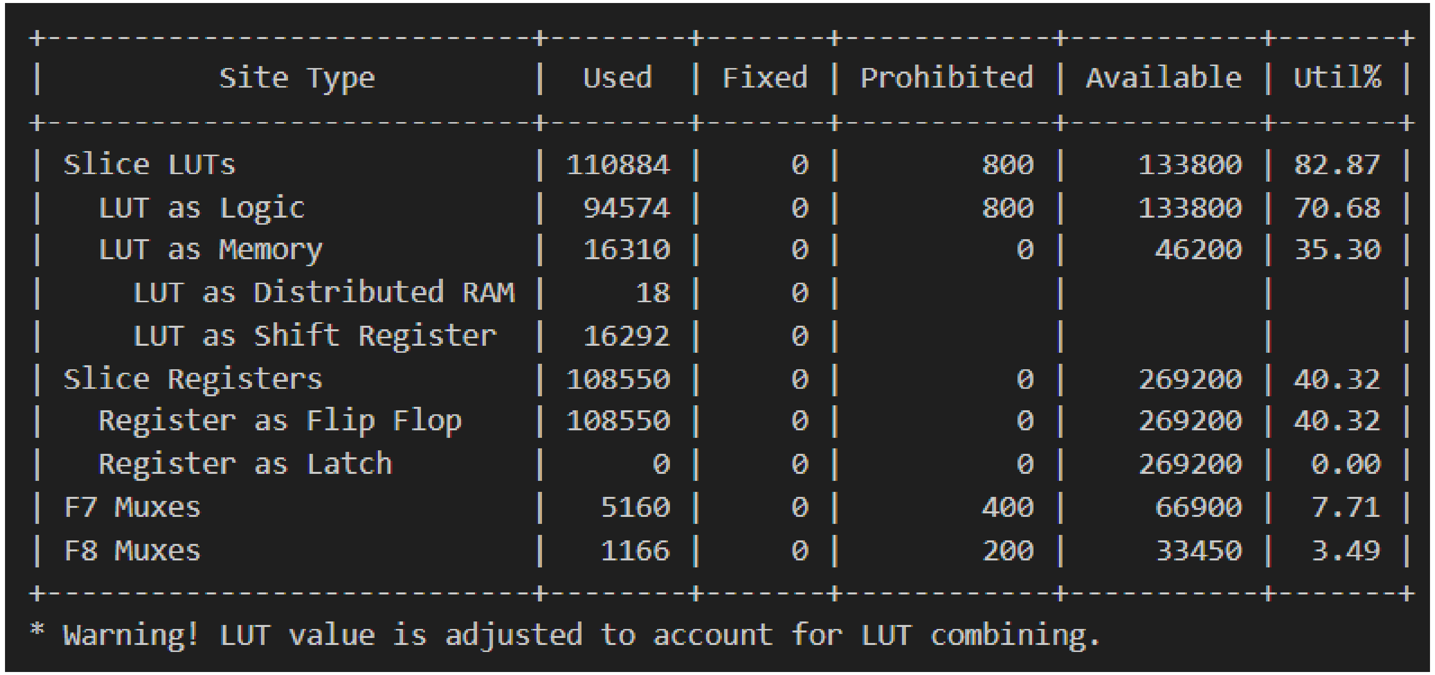
New Timing Issues



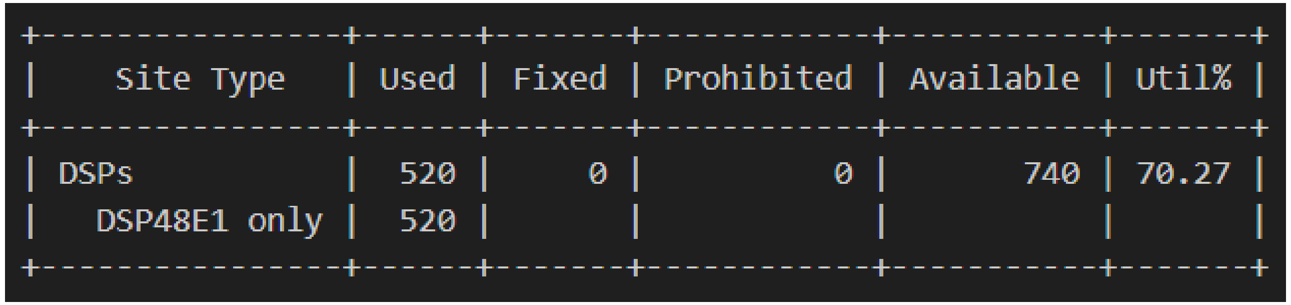
```
------------------------------------------------------------------------------------------------
| Design Timing Summary
| ---------------------
------------------------------------------------------------------------------------------------

    WNS(ns)      TNS(ns)  TNS Failing Endpoints  TNS Total Endpoints      WHS(ns)      THS(ns)
    -------      -------  ---------------------  -------------------      -------      -------
      0.021        0.000                      0               267060        0.012        0.000


All user specified timing constraints are met.
```

- After the MUX and pipelining solution was placed, the merge of all three algorithms was done following this specific branch. However, new timing issues appeared.
- These timing problems were related to a register called packet_size_counter, located in the st40_top module. This register was unnecessary in later commits on the main branch since the usage of the almost_full flag for the FIFOs was already guaranteeing the correct packet size, therefore it was commented and the timing issues were resolved.

Resources Utilization



```
+----------------------------+--------+-------+------------+-----------+-------+
|          Site Type         |  Used  | Fixed | Prohibited | Available | Util% |
+----------------------------+--------+-------+------------+-----------+-------+
| Slice LUTs                 | 110884 |     0 |        800 |    133800 | 82.87 |
|   LUT as Logic             |  94574 |     0 |        800 |    133800 | 70.68 |
|   LUT as Memory            |  16310 |     0 |          0 |     46200 | 35.30 |
|     LUT as Distributed RAM |     18 |     0 |            |           |       |
|     LUT as Shift Register  |  16292 |     0 |            |           |       |
| Slice Registers            | 108550 |     0 |          0 |    269200 | 40.32 |
|   Register as Flip Flop    | 108550 |     0 |          0 |    269200 | 40.32 |
|   Register as Latch        |      0 |     0 |          0 |    269200 |  0.00 |
| F7 Muxes                   |   5160 |     0 |        400 |     66900 |  7.71 |
| F8 Muxes                   |   1166 |     0 |        200 |     33450 |  3.49 |
+----------------------------+--------+-------+------------+-----------+-------+
* Warning! LUT value is adjusted to account for LUT combining.
```



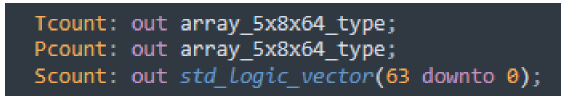
```
+----------------+------+-------+------------+-----------+-------+
|    Site Type   | Used | Fixed | Prohibited | Available | Util% |
+----------------+------+-------+------------+-----------+-------+
| DSPs           |  520 |     0 |          0 |       740 | 70.27 |
|   DSP48E1 only |  520 |       |            |           |       |
+----------------+------+-------+------------+-----------+-------+
```

- There was too much congestion in the initial design, this was produced by the matching trigger, that was previously optimized to not use DSPs.
- After reviewing the DSP usage by milano's filters, some DSPs were freed up in order to implement matching filter's strategy, and exchange its implamantation, so most of EIA's strategy goes to the DSPs, while milano's filters use fabric.
- Congestion was reduced, and DSP usage is not as high, allowing room for more modules.

Timing Issues

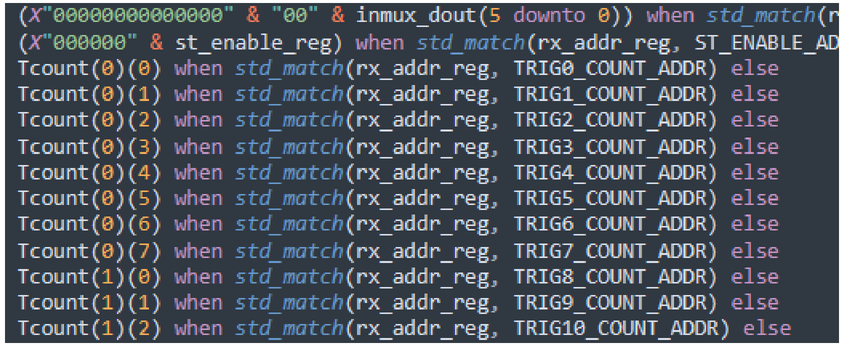
Main branch

Core module



```
Tcount: out array_5x8x64_type;
Pcount: out array_5x8x64_type;
Scount: out std_logic_vector(63 downto 0);
```

DAPHNE top module –BIG mux

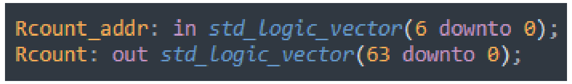


```
(X"00000000000000" & "00" & inmux_dout(5 downto 0)) when std_match(r
(X"000000" & st_enable_reg) when std_match(rx_addr_reg, ST_ENABLE_AD
Tcount(0)(0) when std_match(rx_addr_reg, TRIG0_COUNT_ADDR) else
Tcount(0)(1) when std_match(rx_addr_reg, TRIG1_COUNT_ADDR) else
Tcount(0)(2) when std_match(rx_addr_reg, TRIG2_COUNT_ADDR) else
Tcount(0)(3) when std_match(rx_addr_reg, TRIG3_COUNT_ADDR) else
Tcount(0)(4) when std_match(rx_addr_reg, TRIG4_COUNT_ADDR) else
Tcount(0)(5) when std_match(rx_addr_reg, TRIG5_COUNT_ADDR) else
Tcount(0)(6) when std_match(rx_addr_reg, TRIG6_COUNT_ADDR) else
Tcount(0)(7) when std_match(rx_addr_reg, TRIG7_COUNT_ADDR) else
Tcount(1)(0) when std_match(rx_addr_reg, TRIG8_COUNT_ADDR) else
Tcount(1)(1) when std_match(rx_addr_reg, TRIG9_COUNT_ADDR) else
Tcount(1)(2) when std_match(rx_addr_reg, TRIG10_COUNT_ADDR) else
```

⋮

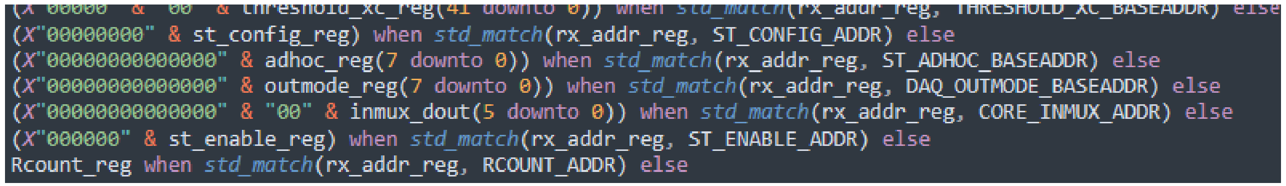
Self-trigger branch

Core module



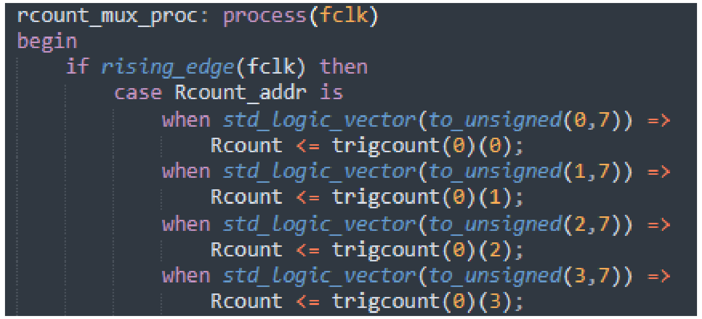
```
Rcount_addr: in std_logic_vector(6 downto 0);
Rcount: out std_logic_vector(63 downto 0);
```

DAPHNE top module –BIG mux



```
(X"00000" & "00" & threshold_xc_reg(41 downto 0)) when std_match(rx_addr_reg, THRESHOLD_XC_BASEADDR) else
(X"00000000" & st_config_reg) when std_match(rx_addr_reg, ST_CONFIG_ADDR) else
(X"00000000000000" & adhoc_reg(7 downto 0)) when std_match(rx_addr_reg, ST_ADHOC_BASEADDR) else
(X"00000000000000" & outmode_reg(7 downto 0)) when std_match(rx_addr_reg, DAQ_OUTMODE_BASEADDR) else
(X"00000000000000" & "00" & inmux_dout(5 downto 0)) when std_match(rx_addr_reg, CORE_INMUX_ADDR) else
(X"000000" & st_enable_reg) when std_match(rx_addr_reg, ST_ENABLE_ADDR) else
Rcount_reg when std_match(rx_addr_reg, RCOUNT_ADDR) else
```

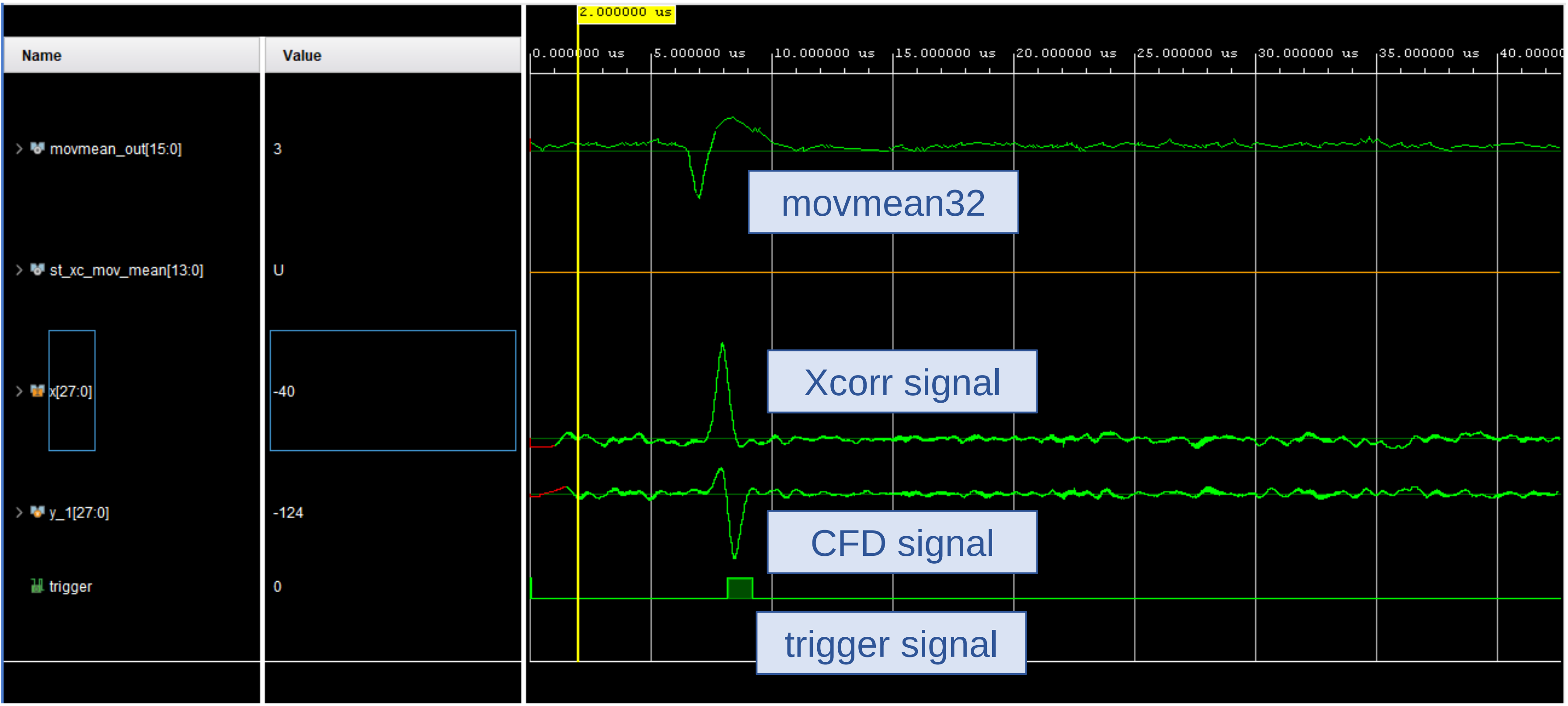
St40_top – synchronous mux to read debug registers



```
rcount_mux_proc: process(fclk)
begin
    if rising_edge(fclk) then
        case Rcount_addr is
            when std_logic_vector(to_unsigned(0,7)) =>
                Rcount <= trigcount(0)(0);
            when std_logic_vector(to_unsigned(1,7)) =>
                Rcount <= trigcount(0)(1);
            when std_logic_vector(to_unsigned(2,7)) =>
                Rcount <= trigcount(0)(2);
            when std_logic_vector(to_unsigned(3,7)) =>
                Rcount <= trigcount(0)(3);
```

- A synchronous mux was inserted in st40_top module. In this way, we reduced the amount of output signals in the core module. Only one signal is used to read the registers at the BIG mux.
- One way to restore compatibility with the DAQ is to include a simple state machine at DAPHNE2 top module that populates registers every 80 clock cycles (there are in total 80 registers). In this way, we can maintain the original read scheme.

Simulations



movmean32

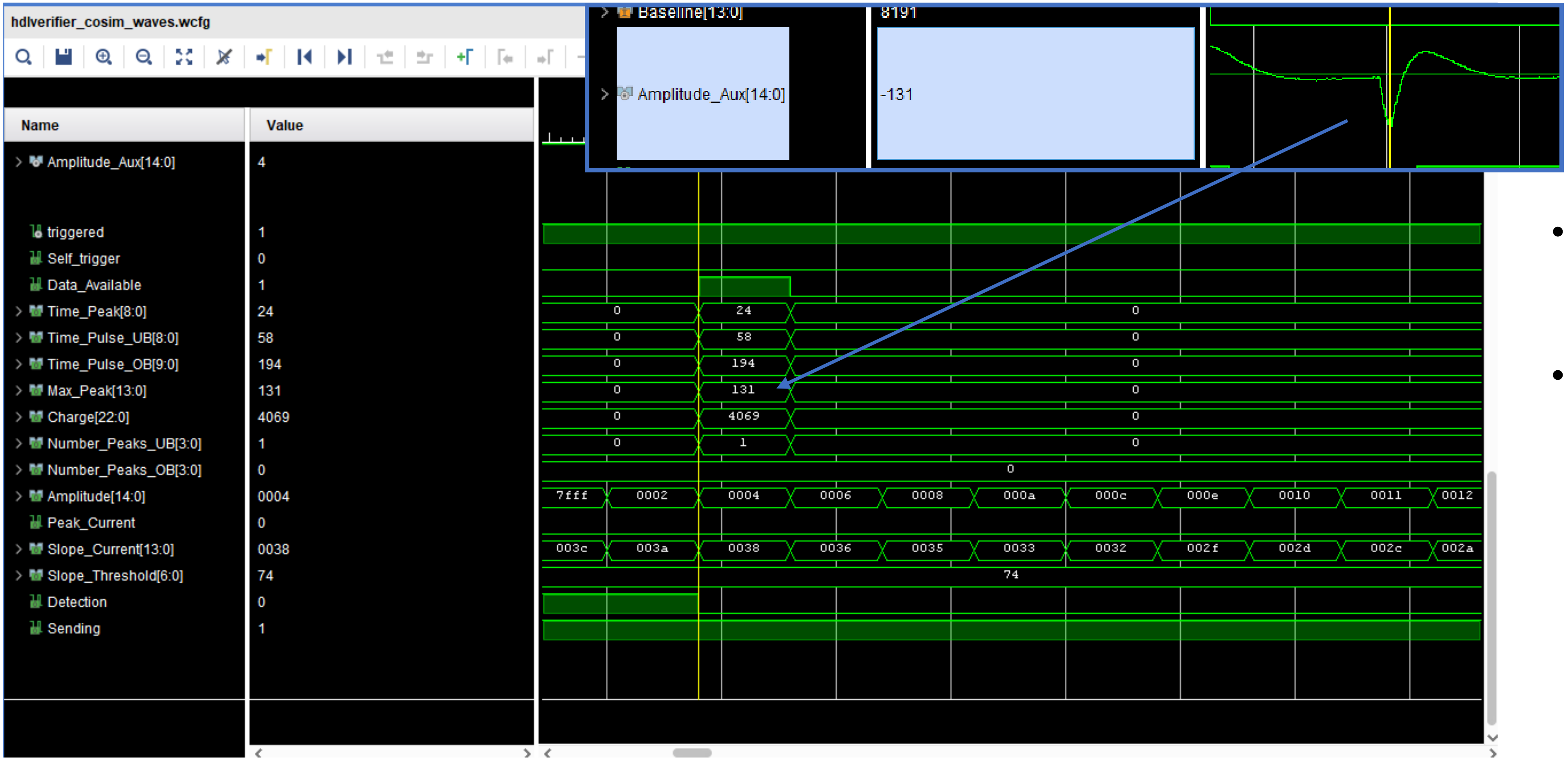
Xcorr signal

CFD signal

trigger signal

- Simulations using FBK waveforms and RAW noise data at VGAIN 0,89V (corresponding to a Dynamic range of around 2000 P.E.) was done to confirm signals are as expected.
- Minor offset and undershoot corrections were introduced and the CFD signal was verified to perform as expected to generate the trigger signal.
- Jitter performance was improved. See following slides.

Simulations



hdlverifier_cosim_waves.wcfg

Name	Value
Amplitude_Aux[14:0]	4
triggered	1
Self_trigger	0
Data_Available	1
Time_Peak[8:0]	24
Time_Pulse_UB[8:0]	58
Time_Pulse_OB[9:0]	194
Max_Peak[13:0]	131
Charge[22:0]	4069
Number_Peaks_UB[3:0]	1
Number_Peaks_OB[3:0]	0
Amplitude[14:0]	0004
Peak_Current	0
Slope_Current[13:0]	0038
Slope_Threshold[6:0]	74
Detection	0
Sending	1

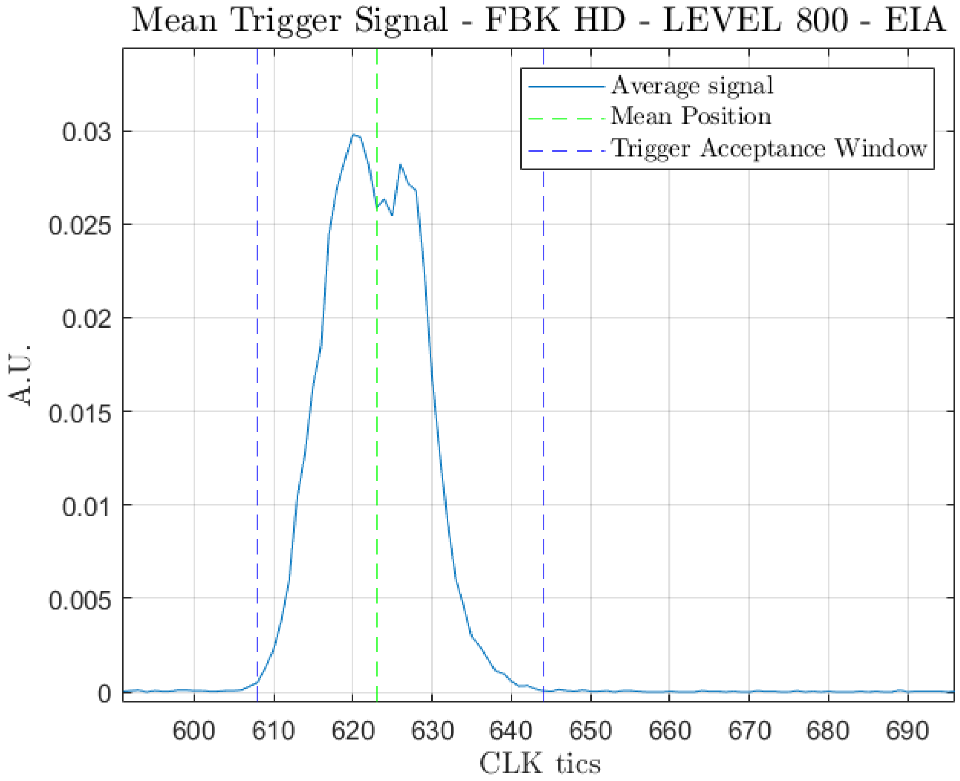
Name	Value
Baseline[13:0]	8191
Amplitude_Aux[14:0]	-131

0 24 0
0 58 0
0 194 0
0 131 0
0 4069 0
0 1 0
0
7fff 0002 0004 0006 0008 000a 000c 000e 0010 0011 0012
003c 003a 0038 0036 0035 0033 0032 002f 002d 002c 002a
74

- Trigger primitives calculation was verified using the simulation.
- The waves file produced by the HDL simulation, shown at the left, verifies that for instance, the peak value is correctly calculated by the trigger primitives calculator.

Simulations

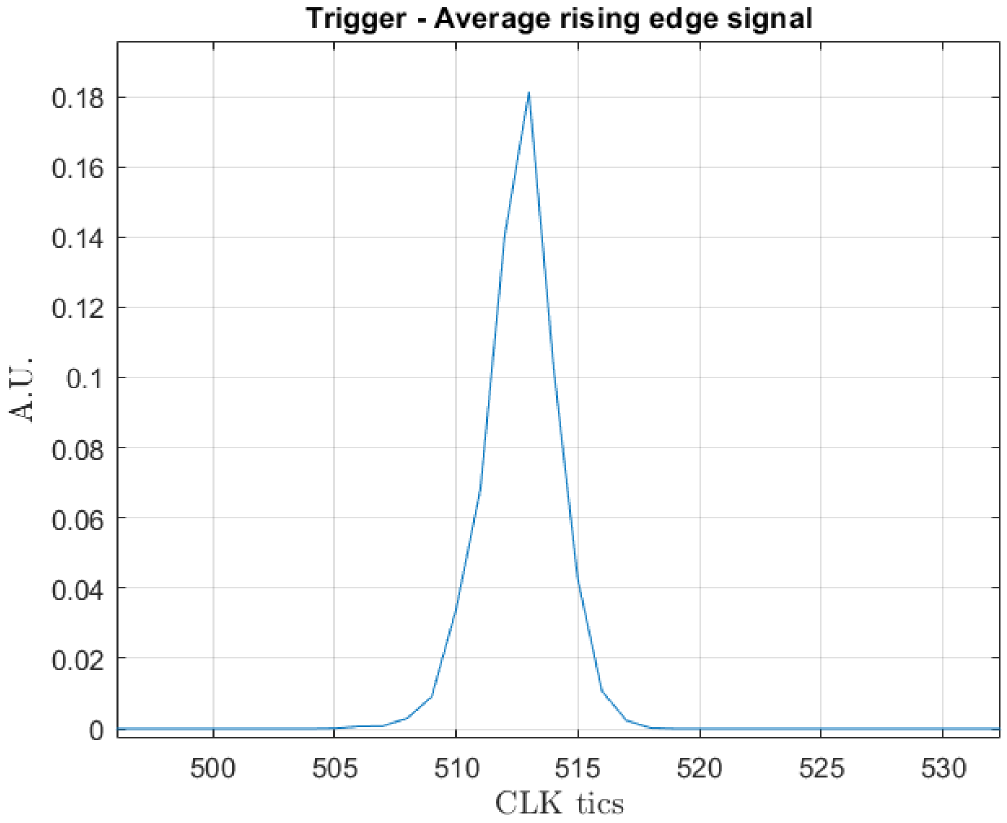
Without CFD - Tested


Mean Trigger Signal - FBK HD - LEVEL 800 - EIA

$$TAW = 36\ tics$$

CFD

With CFD – simulated data


Trigger - Average rising edge signal

$$TAW = 10\ tics$$

- The excess jitter issue observed during the self-trigger test at Milano-Bicocca in the EIA algorithm is mitigated by adding the CFD (constant fraction discriminator) stage.

TAW: *Trigger Acceptance Window*

Conclusions

- The three different algorithms were succesfully integrated:
  - Timing issues were resolved.
  - Incompatibility with DAQ register readout will be adressed.
- Correct behaviour was verified using HDL simulations.
- The CFD module created by milano allowed eia's matching trigger to have an even better jitter behaviour than before.
- Using the OR gate to generate the final trigger output will need more study, since ciemat's jitter behaviour was degraded between the first test and the second test.
- Once the imcompatibility issue is adressed, the design will be ready to be deployed. (More on this depends on experiment status, NP04 will not be available for testing, there is a small possibility of testing in NP02, else, we will start the migration to DAPHNE V3)