Trigger Primitives calculation in DAPHNE 2A

Antonio Verdugo on behalf of CIEMAT Team

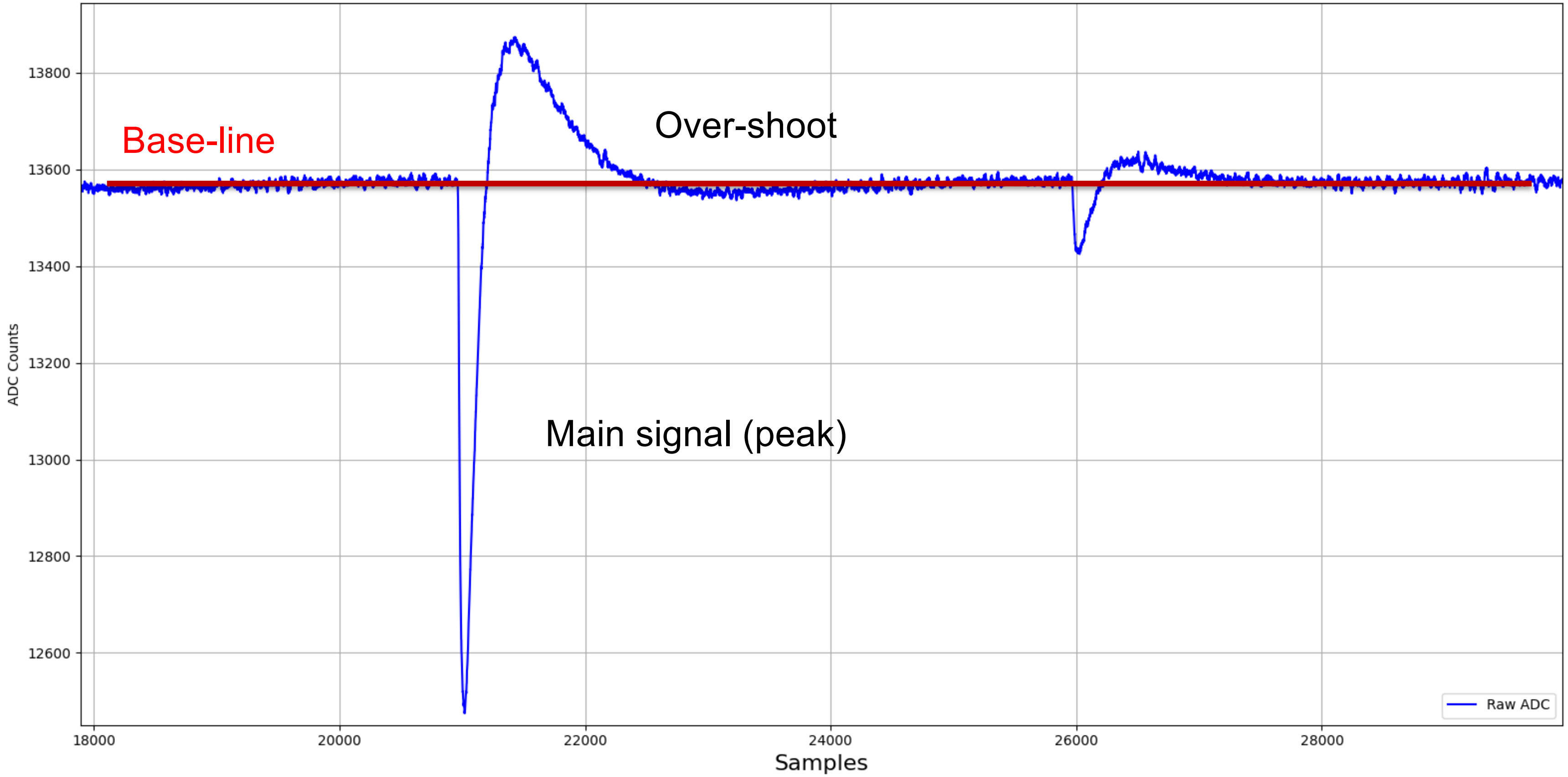
PDS Data Taking Meeting 1st February 2024

Goal

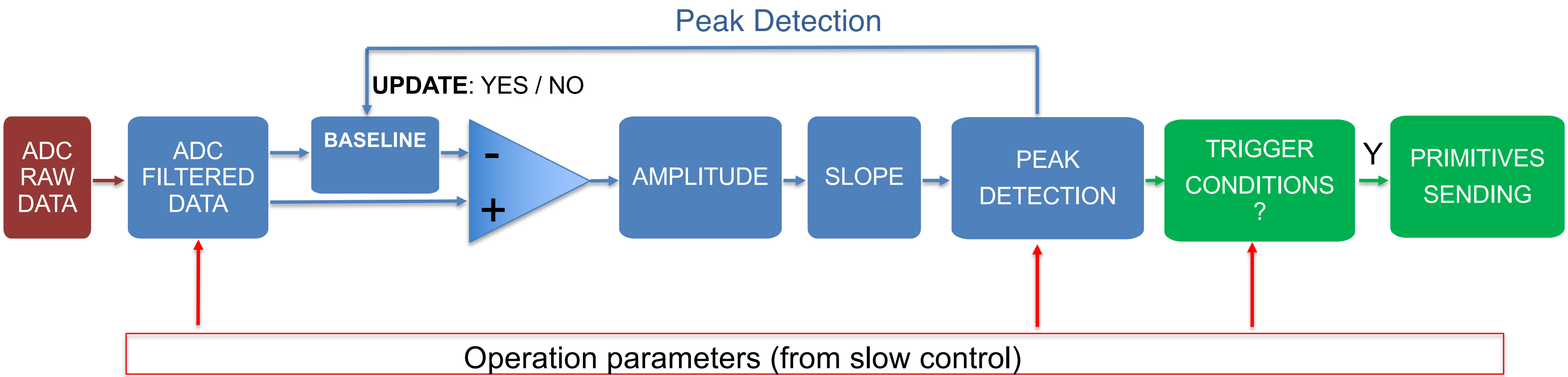
➢ **Develop, implement, and test real time trigger primitives (TP) algorithms in the DAPHNE FPGA.**

- **Intermediate steps:**
  - **Understand the functionality of Daphne HW/FW.**
  - **Have a Daphne V2A and learn how to operate it.**
  - **Implement a self-trigger algorithm (not available when we started) in order to trigger the TP calculation.**

Typical light signal in DAPHNE



Self-trigger & Primitive Calculation ALGORITHM

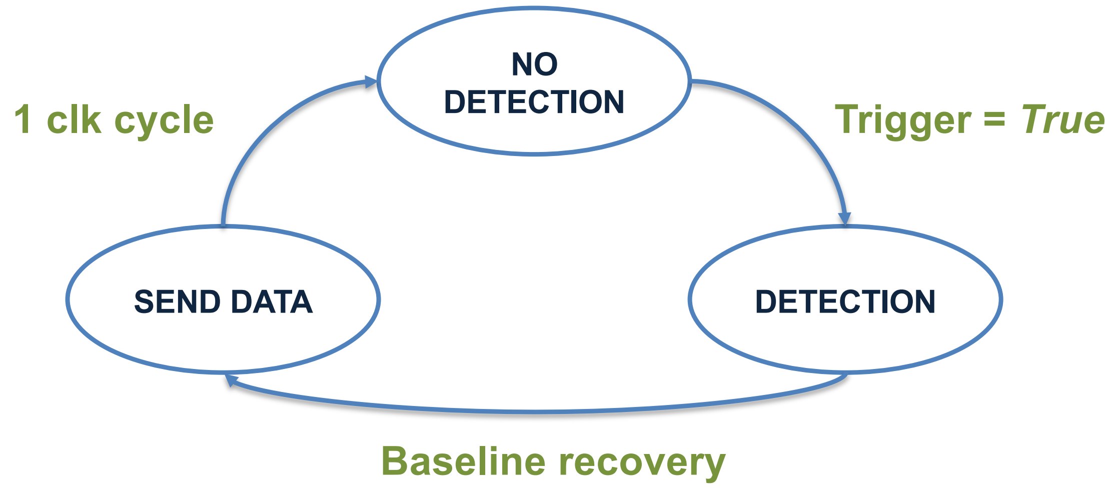


- **ADC FILTERED DATA**: Moving average of 2 ADC RAW DATA SAMPLES → Reduces High Frequency noise.
- **BASELINE:**
  - Based on cumulative average over previous N (4) samples.
  - Stop baseline calculation if peak is detected.
- **AMPLITUDE =** Filtered Data – Baseline.
- **SLOPE =** Amplitude last simple – Amplitude previous sample.
- **PEAK DETECTION**: Threshold over the slope.
- **TRIGGER CONDITIONS**: Based on the TP: Amplitude, charge, ..

Updated on each CLK cycle

Primitive Calculation: First Approach

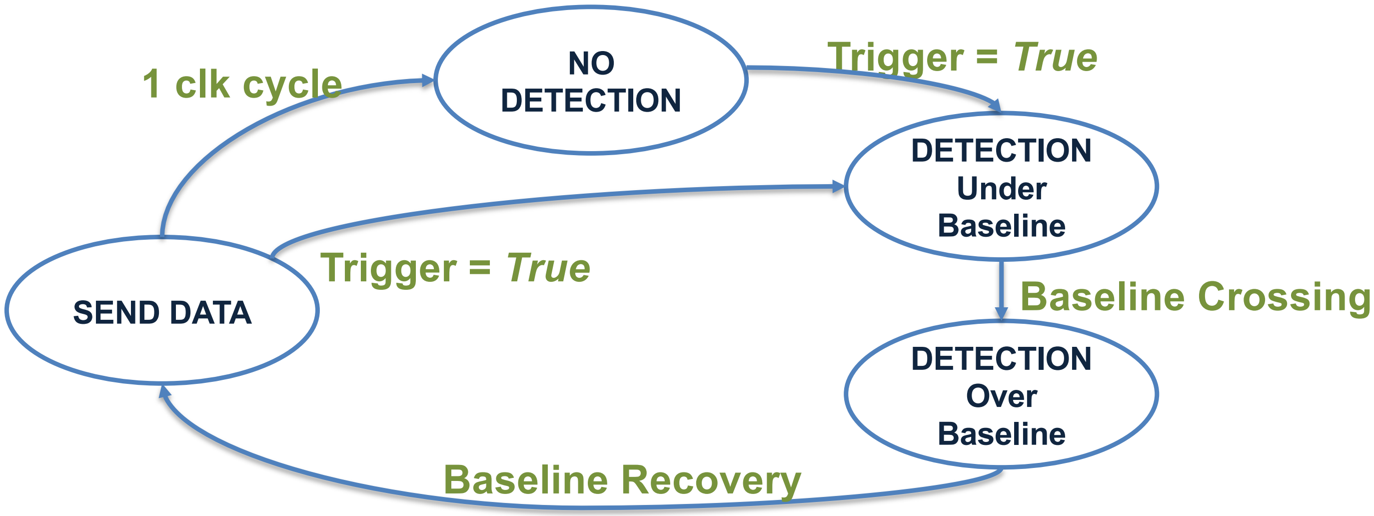
Trigger allowed on the over-shoot



- **NO DETECTION *State*:**
  - Peak detection variables calculation (Baseline Calculation).
- **TRIGGER CONDITION**: When a Peak fullfilling trigger conditions is detected in NO DETECTION *State.*
- **DETECTION *State* :**
  - Peak detection variables calculation (Baseline remains constant)
  - Waveform`s Primitive Calculation.
  - Peak detection does not generate a self-trigger signal.
- **SEND DATA *State*:** Waveform’s Primitve Data available.

Primitive Calculation: Second Approach

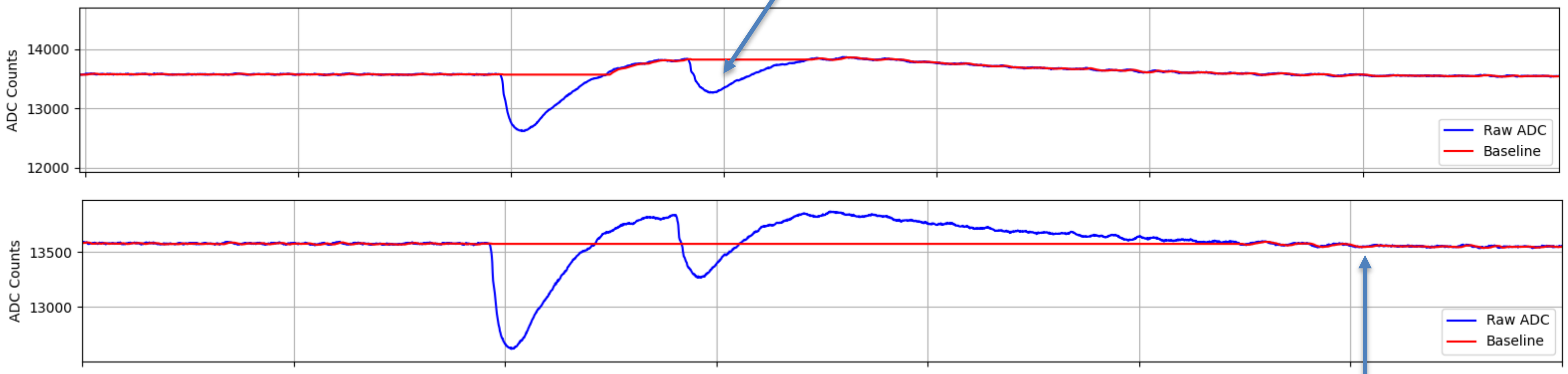
Trigger not allowed during over-shoot



- **NO DETECTION *State*:**
  - Peak detection variables calculation (Baseline Calculation).
- **TRIGGER CONDITION**: When a Peak fullfilling trigger conditions is detected in NO DETECTION *State.*
- **DETECTION *States* :**
  - Peak detection variables calculation (Baseline remains constant)
  - Waveform`s Primitive Calculation.
  - Peak detection does not generate a self-trigger signal.
- **SEND DATA *State*:** Waveform's Primitve Data available.

Comparison: FIRST vs SECOND approaches

Second Peak detected -> Additional trigger & primitives calculation



Peak detection disabled until base-line recovery

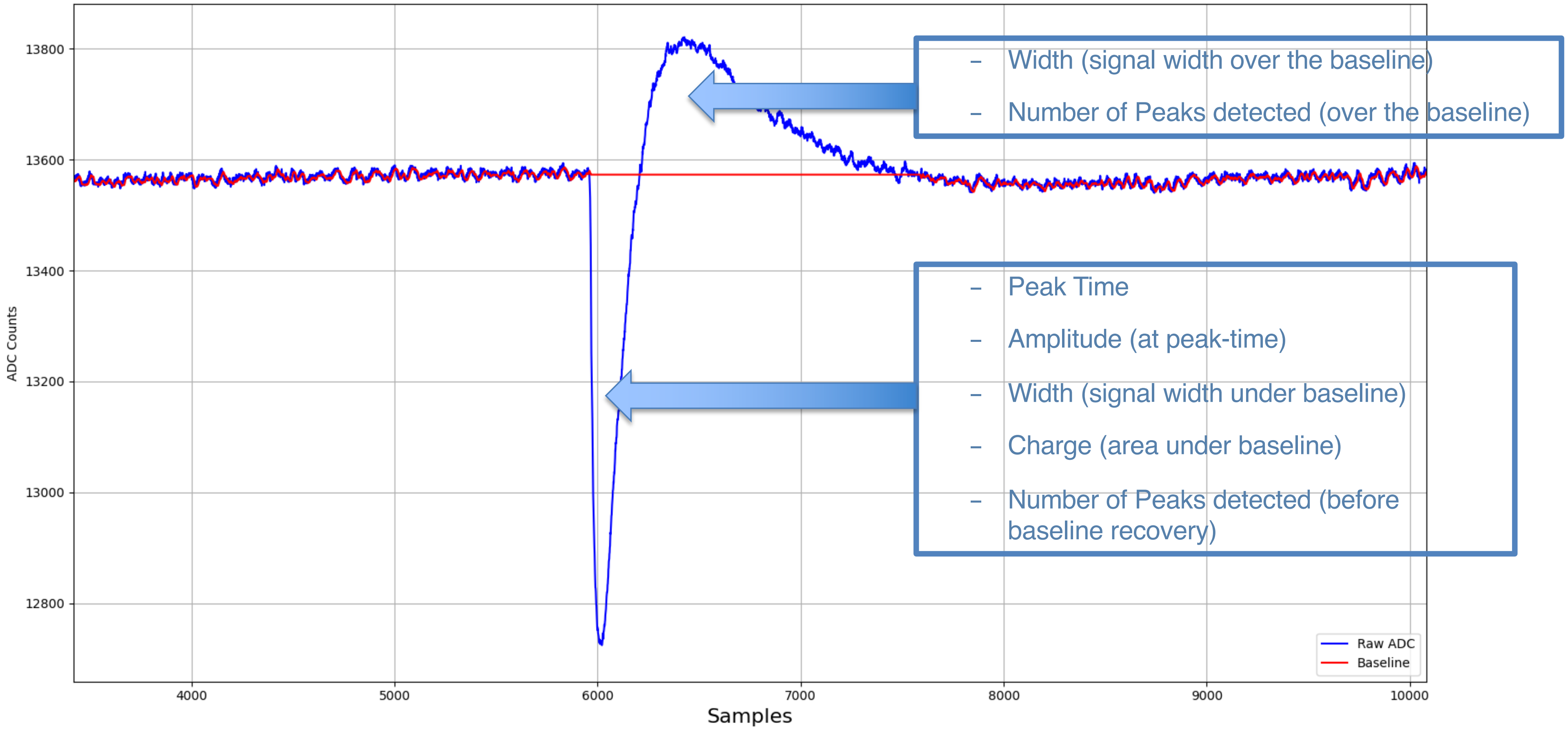
MAIN DIFFERENCES

- **Baseline:**
  - *OLD APPROACH: Follows undershoot.*
  - *NEW APPROACH: Remains constant.*
- **Self-Trigger:**
  - *OLD APPROACH: It is allowed during undershoot.*
  - *NEW APPROACH: It is not allowed during undershoot.*

MOTIVATION

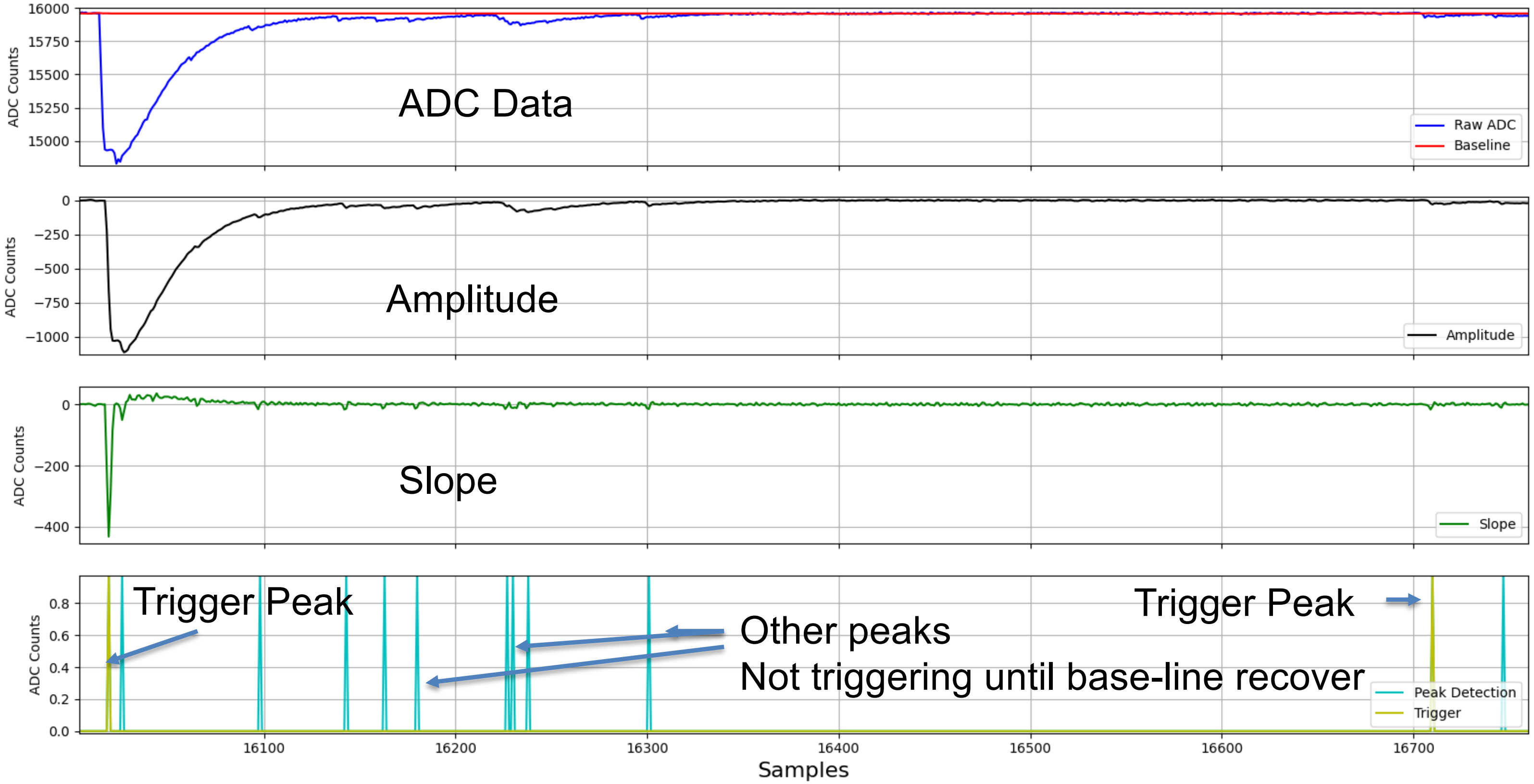
- Self-Trigger Event → Waveform's Primitive Calculation.
- Waveform`s Primitive Calculation is not accurate in the undershoot.

Trigger Primitives (last proposal)



Example waveform and results

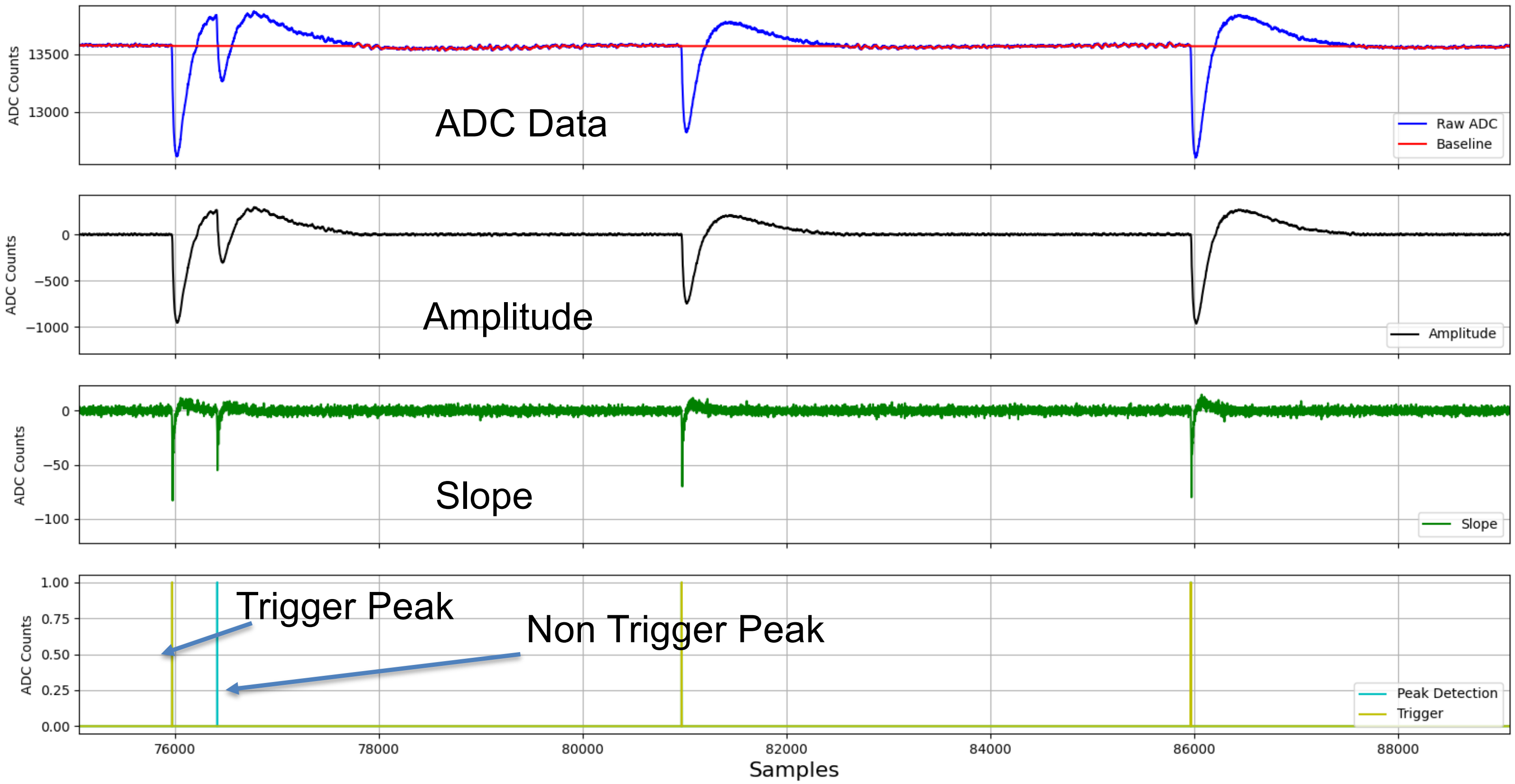
Post-synthesis timing simulation (SiPM signal from alpha source)




```
---------- DETECTED PULSE  4 ----------
Time to Peak      =  8
Pulse Width       =  343
Max Amplitude     =  -1114
Charge                =  -50108
Number of Peaks       =  10
---------- DETECTED PULSE  5 ----------
Time to Peak      =  6
Pulse Width       =  97
Max Amplitude     =  -27
Charge                =  -1164
Number of Peaks       =  2
```

Example waveform and results

Post-synthesis timing simulation (MegaCell)




```
--------- DETECTED PULSE  15 ---------
Time to Peak     =  53
Pulse Width      =  249
Max Amplitude    =  952
Charge               =  119668
Number of Peaks      =  1
Undershoot Width     =  1526
Number of Peaks in Undershoot   =  1
--------- DETECTED PULSE  16 ---------
Time to Peak     =  48
Pulse Width      =  243
Max Amplitude    =  745
Charge               =  91433
Number of Peaks      =  1
Undershoot Width     =  1398
Number of Peaks in Undershoot   =  0
--------- DETECTED PULSE  17 ---------
Time to Peak     =  54
Pulse Width      =  245
Max Amplitude    =  961
Charge               =  120948
Number of Peaks      =  1
Undershoot Width     =  1377
Number of Peaks in Undershoot   =  0
```

Space required for Trigger Primitives sending

```
Data_Available: out std_logic;                     -- Primitives calculation available. Active HIGH
Time_Peak: out  std_logic_vector(7 downto 0);      -- Time in Samples to achieve de Max peak
Time_Pulse_UB: out  std_logic_vector(9 downto 0);   -- Time in Samples of the light pulse signal is UNDER BASELINE (without undershoot)
Time_Pulse_OB: out  std_logic_vector(10 downto 0);  -- Time in Samples of the light pulse signal is OVER BASELINE (undershoot)
Max_Peak: out  std_logic_vector(15 downto 0);       -- Amplitude in ADC counts od the peak
Charge: out  std_logic_vector(19 downto 0);         -- Charge of the light pulse (without undershoot) in ADC*samples
Number_Peaks_UB: out  std_logic_vector(3 downto 0);-- Number of peaks detected when signal is UNDER BASELINE (without undershoot).
Number_Peaks_OB: out  std_logic_vector(3 downto 0);-- Number of peaks detected when signal is OVER BASELINE (undershoot).
```

73 bits required vs 32 bits reserved in the trailer word

We could start with a reduced number of Trigger Primitives for the ProtoDune HD in order to test, at least, the TP calculation and the comunication with the DAQ.

Status and Next Steps

- The algorithm was fully tested with Python scripts using real data from CIEMAT´s PDE measurement setups.

- One channel “Self-Trigger & Primitive Calculation” block is implemented and simulated (Post-Synthesis timing simulation) with Vivado.

- We have also tested one channel block in Daphne v2A firmware by means of spy-buffers.

Next:

- Implement and test the block on all 40 channels using the actual output FIFOs and collecting data from the Daphne output frame. As a DAQ (Felix) is required to read the output frame, this test must be performed where a DAQ is available (CERN, FERMILAB ?).

Thanks for your attention!

Peak Detection - BACKUP

Baseline calculation is based in calculating cumulative average.

$$\bar{x} = \frac{\sum x_i}{n} \rightarrow \overline{x_{i+1}} = \bar{x}_i + \frac{x_{i+1} - \bar{x}_i}{n+1}$$

$$\overline{x_{i+1}} = \bar{x}_i + \frac{x_{i+1} - \bar{x}_i}{2^N}$$

FILTERED DATA	BASELINE	AMPLITUDE	SOLPE	PEAK DETECTION
Initial condition: $F_0 = x_0$ **Algorithm:** $F_{i+1} = \frac{x_i + x_{i+1}}{2}$	**Initial condition:** $B_0 = x_0$ **Algorithm:** **If** $Detection$ $B_{i+1} = B_i$ **Else** $B_{i+1} = B_i + \frac{F_{i+1} - B_i}{8}$	**Initial condition:** $A_0 = 0$ **Algorithm:** $A_{i+1} = F_{i+1} - B_{i+1}$	**Initial condition:** $S_0 = 0$ **Algorithm:** $S_{i+1} = A_{i+1} - A_i$	**Initial condition:** $P_0 = false$ **Algorithm:** **If** $S_{i+1} < -10$ $P_0 = false$ **Else** $P_0 = true$

Waveform's Primitive Calculation- BACKUP

While **DETECTION** ***State***

PULSE WITH	TIME TO PEAK	MAX AMPLITUDE	CHARGE	NUMBER OF PEAKS
Initial condition: $W_0 = 0$ **Algorithm:** $W_{i+1} = W_i + 1$	**Initial condition:** $TP_0 = 0$ **Algorithm:** **If** $Amplitude_{i+1} < MA_i$ $T_{i+1} = W_{i+1}$ **Else** $T_{i+1} = T_i$	**Initial condition:** $MA_0 = 0$ **Algorithm:** **If** $Amplitude_{i+1} < MA_i$ $MA_{i+1} = Amplitude_{i+1}$ **Else** $MA_{i+1} = MA_i$	**Initial condition:** $C_0 = 0$ **Algorithm:** $C_{i+1} = C_i + Amplitude_{i+1}$	**Initial condition:** $NP_0 = 0$ **Algorithm:** **If** $Peak\_Detection$ $NP_{i+1} = NP_i + 1$ **Else** $NP_{i+1} = NP_i$