Daphne v2A at CIEMAT

Antonio Verdugo, **Ignacio López de Rego**
CIEMAT

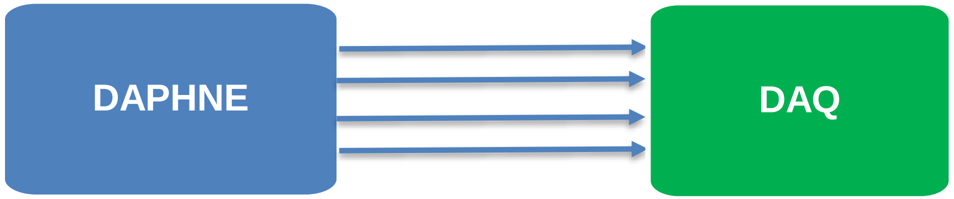
Warm Electronics Meeting 26th October 2023

Goals

- First goal – FPGA Design:
  - Design and Implement a Self-Trigger algorithm.
  - Design and Implement a Waveform`s Primitive Calculation algorithm.
  - Minimize, as much as possible, HW resources used in the FPGA.
- Second goal - Adapt Daphne for PMTs data acquisition:

Self-trigger & Primitive Calculation ALGORITHM (1)

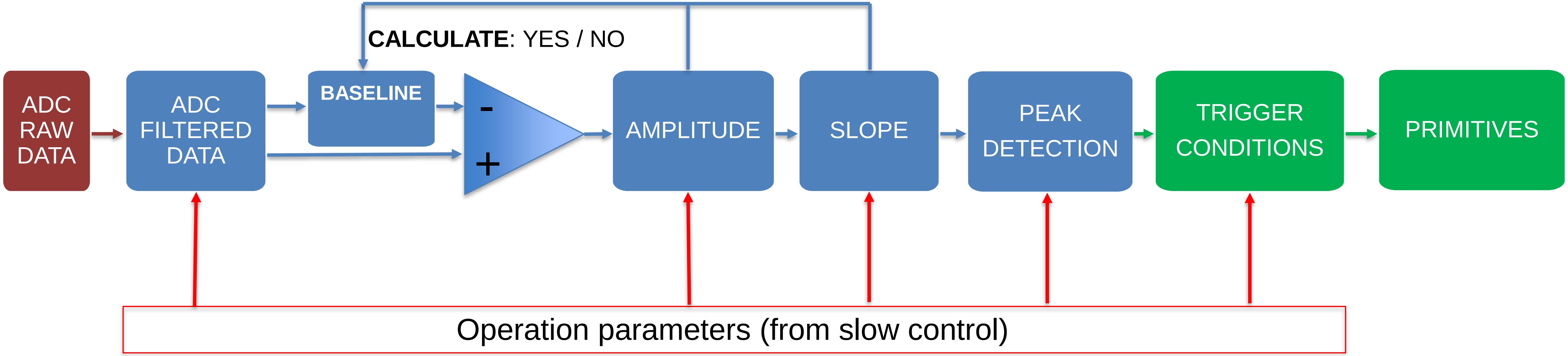
Motivation



- **Self-trigger:**
  - Only packages with useful information are sent → Reducing data flow compared with streaming mode.
- **Basic Waveform's Primitive Calculation:**
  - Basic information of the signal is sent in the headers of the package.
  - Global PDS trigger can be done based on that specific information → At an early stage it is not necessary to analyze every single waveform at the DAQ.

Self-trigger & Primitive Calculation ALGORITHM (2)

Peak Detection

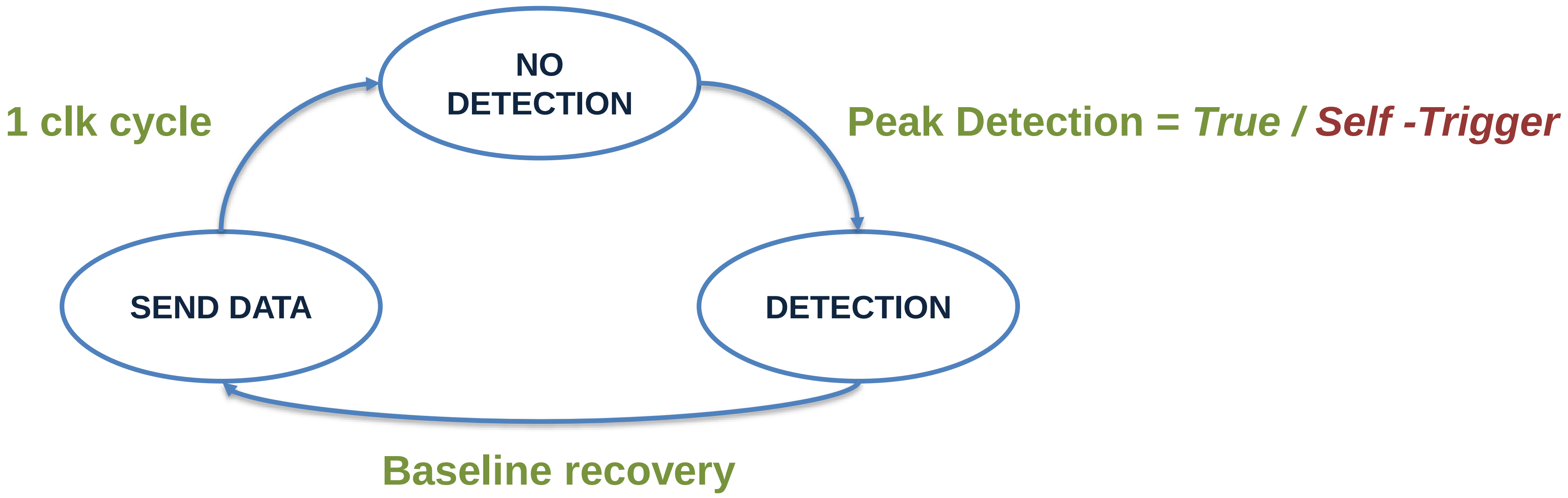


- **ADC FILTERED DATA**: Moving average of 2 ADC RAW DATA SAMPLES → Reduces High Frequency noise.
- **BASELINE:**
  - Based on cumulative average over previous N (4) samples.
  - Stop baseline calculation if peak is detected.
- **AMPLITUDE =** Filtered Data – Baseline.
- **SLOPE =** Amplitude last simple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

Actualized value each CLK cycle

Self-trigger & Primitive Calculation ALGORITHM (3)

Trigger Condition & Primitive Calculation

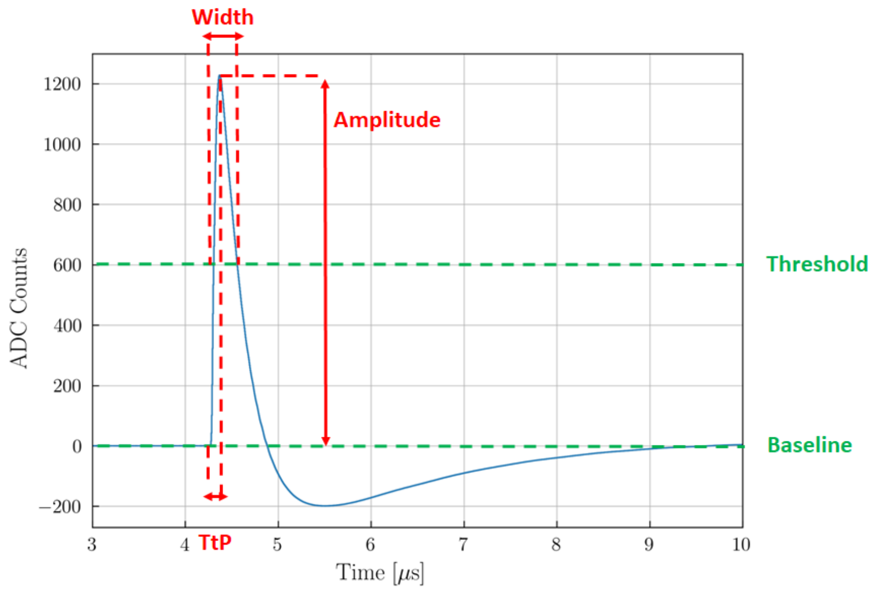


- **TRIGGER CONDITION**: When a Peak is detected in NO DETECTION *State.*
- **NO DETECTION *State*:**
  - Peak detection variables calculation (Baseline Calculation).
- **DETECTION *State* :**
  - Peak detection variables calculation (Baseline remains constant)
  - Waveform`s Primitive Calculation.
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

Self-trigger & Primitive Calculation ALGORITHM (4)

Waveform's Primitives

- **Real-time calculation** of different parameters (trigger primitives) from the SiPM's waveform:
  - Peak Time
  - Amplitude (at peak-time)
  - Width (signal width above threshold)
  - Charge (area above baseline)
  - Number of Peaks detected (before baseline recovery)



- **In order to reduce HW resources (Raw units)**
  - Time (Peak Time, Width): Relative to trigger timestamp, number of bins / samples.
  - Amplitude: ADC counts
  - Charge: ADC*tics

Self-trigger & Primitive Calculation ALGORITHM (5)

Achievements and future plans

Achievements

- Design a Self-Trigger & Primitive Calculation Algorithm.
- Test the algorithm with Python Scripts using real data from CIEMAT´s PDE measurement setups.
- Implement and Simulate (Post-Synthesis timing simulation) a 1 channel “Self-Trigger & Primitive Calculation” block in the FPGA
  - HW Resources: **246 LUT and 247 FF**
- Test the 1 channel “Self-Trigger & Primitive Calculation” block in Daphne v2A firmware.
  - The block was placed in Daphne TOP Module just to test functionality. Several Spybuffers were created in order to follow different algorithm variables (Baseline, Amplitude, peak detections…)

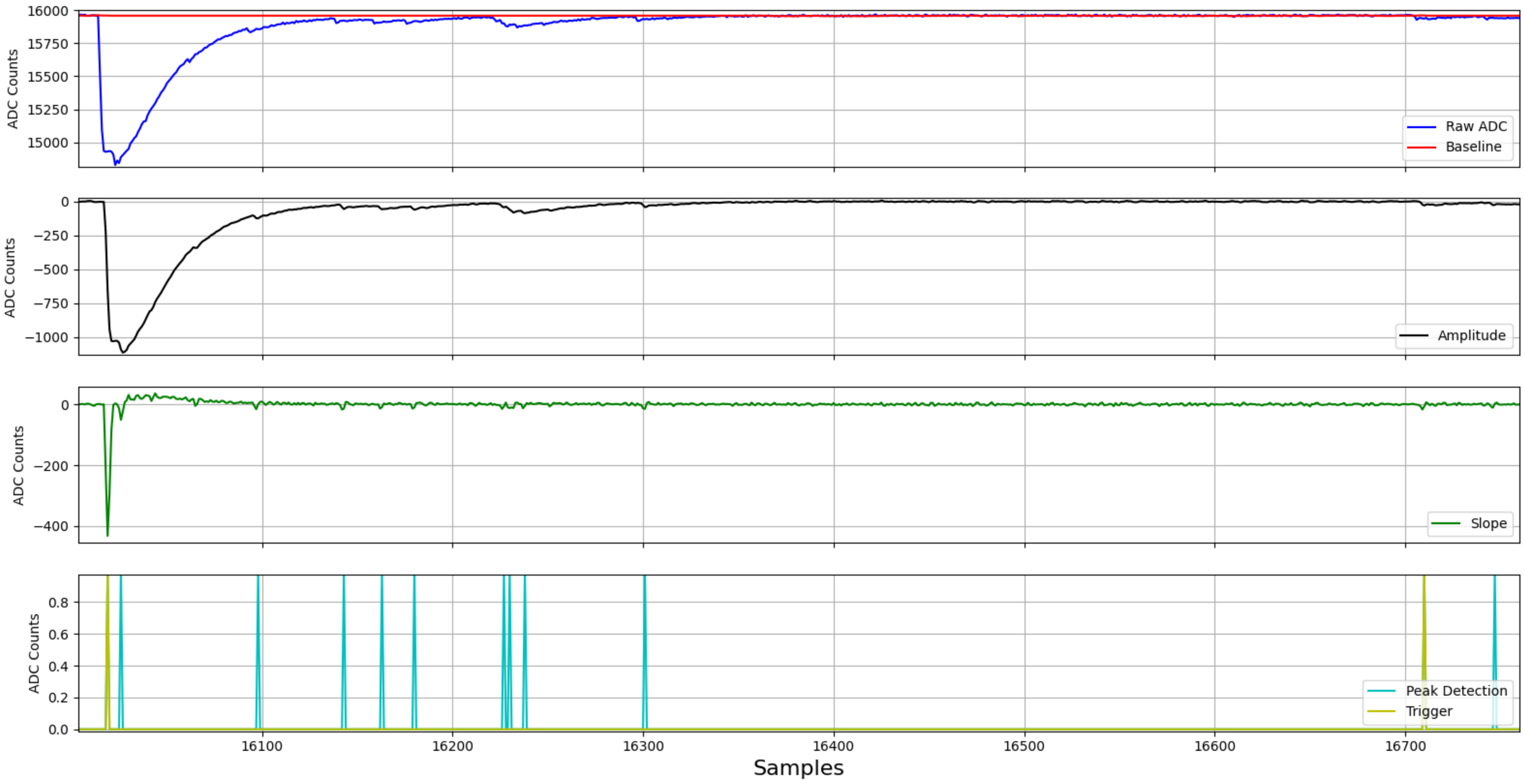
Future plans

- Test the “Self-Trigger & Primitive Calculation” using the real output FIFOs and collecting data from Daphne Streaming frame format → It is required to add extra header to fit all waveform’s primitives
- Implement a block to check trigger coincidences between channels.

<table>
<tr><th></th><th>K/D</th><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td>0</td><td colspan="6">Link</td><td colspan="4">Slot</td><td colspan="10">CrateID</td><td colspan="6">DetID</td><td colspan="6">Version</td></tr>
<tr><td>1</td><td>0000</td><td colspan="32">Timing master Time stamp [31:0]</td></tr>
<tr><td>2</td><td>0000</td><td colspan="32">Timing master Time stamp [63:32]</td></tr>
<tr><td>3</td><td>0000</td><td colspan="26">TBD</td><td colspan="6">Channel</td></tr>
<tr><td>4</td><td>0000</td><td colspan="32">TBD</td></tr>
<tr><td>5</td><td>0000</td><td colspan="4">ADC([3:0],2)</td><td colspan="14">ADC([13:0],1)</td><td colspan="14">ADC([13:0],0)</td></tr>
<tr><td>6</td><td>0000</td><td colspan="8">ADC([7:0],4)</td><td colspan="14">ADC([13:0],3)</td><td colspan="10">ADC([13:4],2)</td></tr>
<tr><td>7</td><td>0000</td><td colspan="12">ADC([11:0],6)</td><td colspan="14">ADC([13:0],5)</td><td colspan="6">ADC([13:8],4)</td></tr>
<tr><td>8</td><td>0000</td><td colspan="2">ADC([1:0],9)</td><td colspan="14">ADC([13:0],8)</td><td colspan="14">ADC([13:0],7)</td><td colspan="2">ADC([13:12],6)</td></tr>
<tr><td>9</td><td>0000</td><td colspan="6">ADC([5:0],11)</td><td colspan="14">ADC([13:0],10)</td><td colspan="12">ADC([13:2],9)</td></tr>
<tr><td>10</td><td>0000</td><td colspan="10">ADC([9:0],13)</td><td colspan="14">ADC([13:0],12)</td><td colspan="8">ADC([13:6],11)</td></tr>
<tr><td>11</td><td>0000</td><td colspan="14">ADC([13:0],15)</td><td colspan="14">ADC([13:0],14)</td><td colspan="4">ADC([13:10],13)</td></tr>
<tr><td>...</td><td>0000</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>446</td><td>0000</td><td colspan="4">ADC([3:0],1010)</td><td colspan="14">ADC([13:0],1009)</td><td colspan="14">ADC([13:0],1008)</td></tr>
<tr><td>447</td><td>0000</td><td colspan="8">ADC([7:0],1012)</td><td colspan="14">ADC([13:0],1011)</td><td colspan="10">ADC([13:4],1010)</td></tr>
<tr><td>448</td><td>0000</td><td colspan="12">ADC([11:0],1014)</td><td colspan="14">ADC([13:0],1013)</td><td colspan="6">ADC([13:8],1012)</td></tr>
<tr><td>449</td><td>0000</td><td colspan="2">ADC([1:0],1017)</td><td colspan="14">ADC([13:0],1016)</td><td colspan="14">ADC([13:0],1015)</td><td colspan="2">ADC([13:12],1014</td></tr>
<tr><td>450</td><td>0000</td><td colspan="6">ADC([5:0],1019)</td><td colspan="14">ADC([13:0],1018)</td><td colspan="12">ADC([13:2],1017)</td></tr>
<tr><td>451</td><td>0000</td><td colspan="10">ADC([9:0],1021)</td><td colspan="14">ADC([13:0],1020)</td><td colspan="8">ADC([13:6],1019)</td></tr>
<tr><td>452</td><td>0000</td><td colspan="14">ADC([13:0],1023)</td><td colspan="14">ADC([13:0],1022)</td><td colspan="4">ADC([13:10],1021)</td></tr>
</table>

Self-trigger & Primitive Calculation ALGORITHM (6)

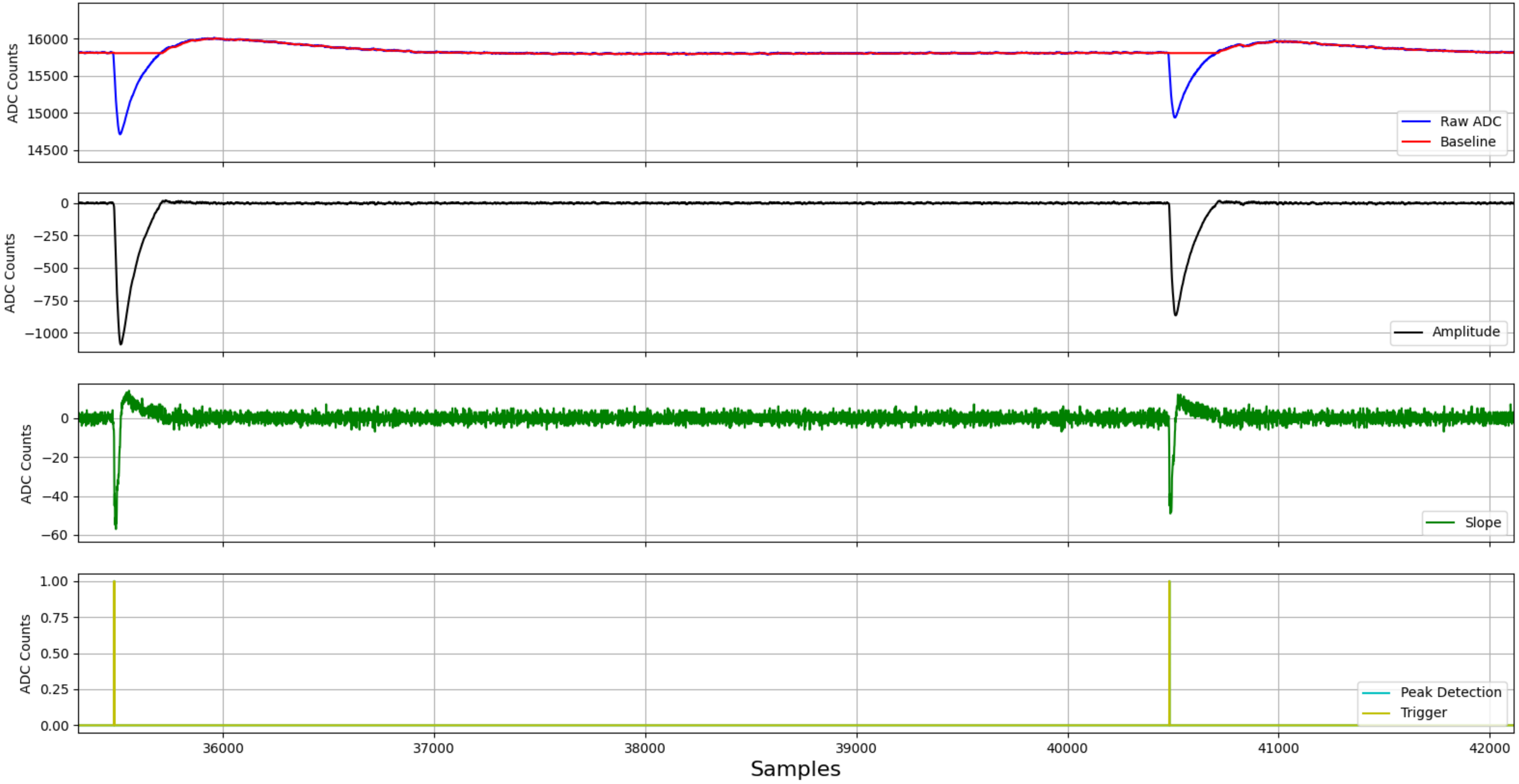
Post-synthesis timing simulation → SiPM




```
---------- 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
```

Self-trigger & Primitive Calculation ALGORITHM (7)

Post-synthesis timing simulation → X-Arapuca (SuperCell)

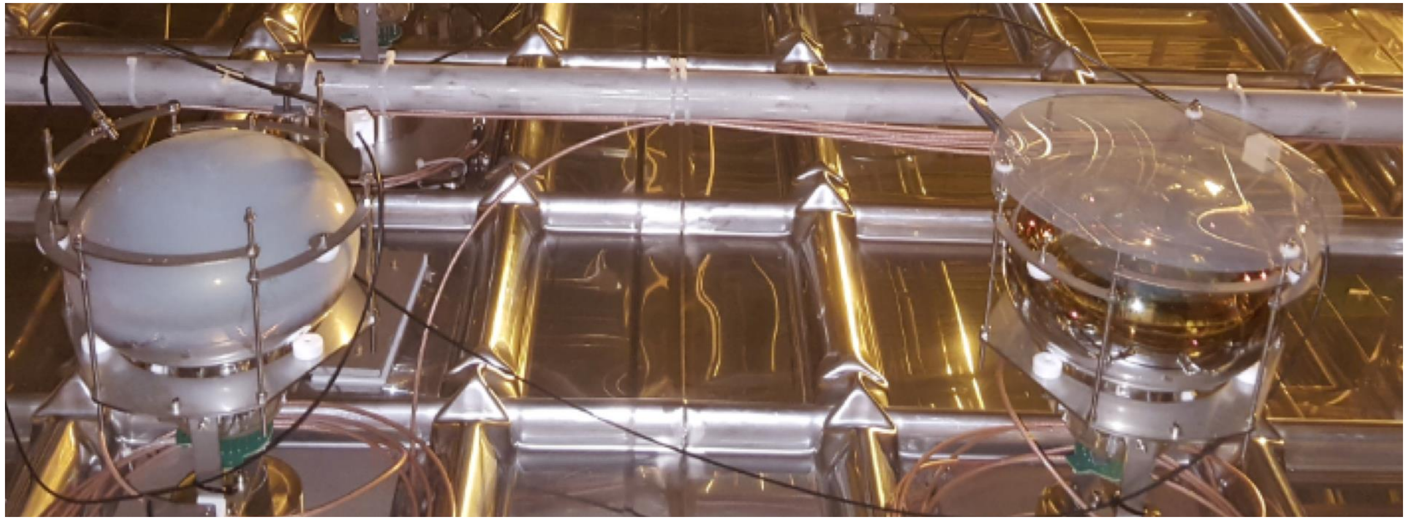



```
---------- DETECTED PULSE  8 ----------
Time to Peak      =  30
Pulse Width       =  222
Max Amplitude     =  -1093
Charge                =  -102784
Number of Peaks       =  1
---------- DETECTED PULSE  9 ----------
Time to Peak      =  30
Pulse Width       =  226
Max Amplitude     =  -868
Charge                =  -80940
Number of Peaks       =  1
```

Adapt Daphne for PMTs data acquisition (1)

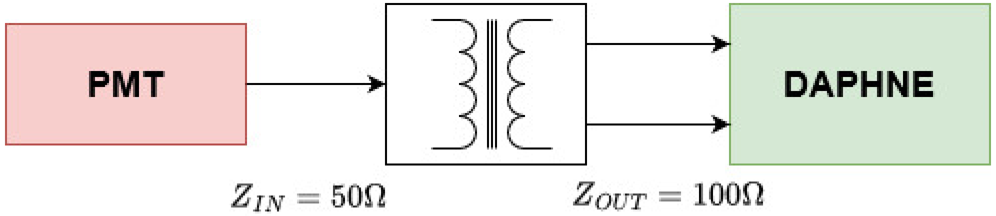
Motivation

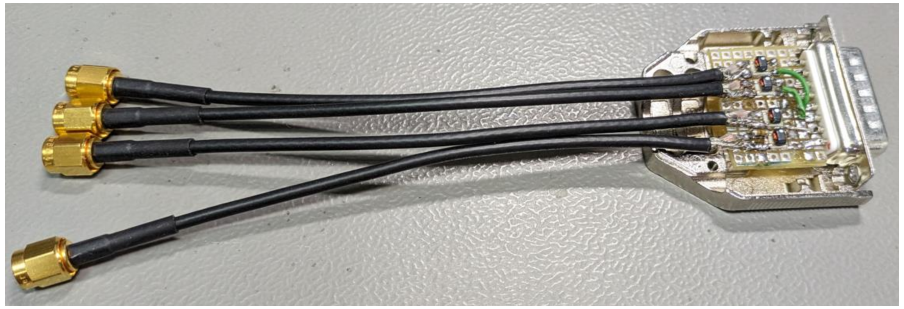
- **Proposal for installing 24 PMTs from ProtoDUNE-DP in ProtoDUNE-VD.**
- **An interface module between the PMT signals and Daphne will be required for signal conversion.**
  - **Impedance and signal conversion is required from single-ended to differential.**
  - **If required, we could also implement a trigger based on the coincidence of several PMTs.**



Adapt Daphne for PMTs data acquisition (2)

Interface Daphne - PMT





- **Protoboard →** Transformer
- **Transformer: Mini-Circuits TC2-1T+ →** Reduces crosstalk between channels
  - Single-ended to Differential
  - Impedance conversion

Adapt Daphne for PMTs data acquisition (3)

Achievements and future plans

Achievements

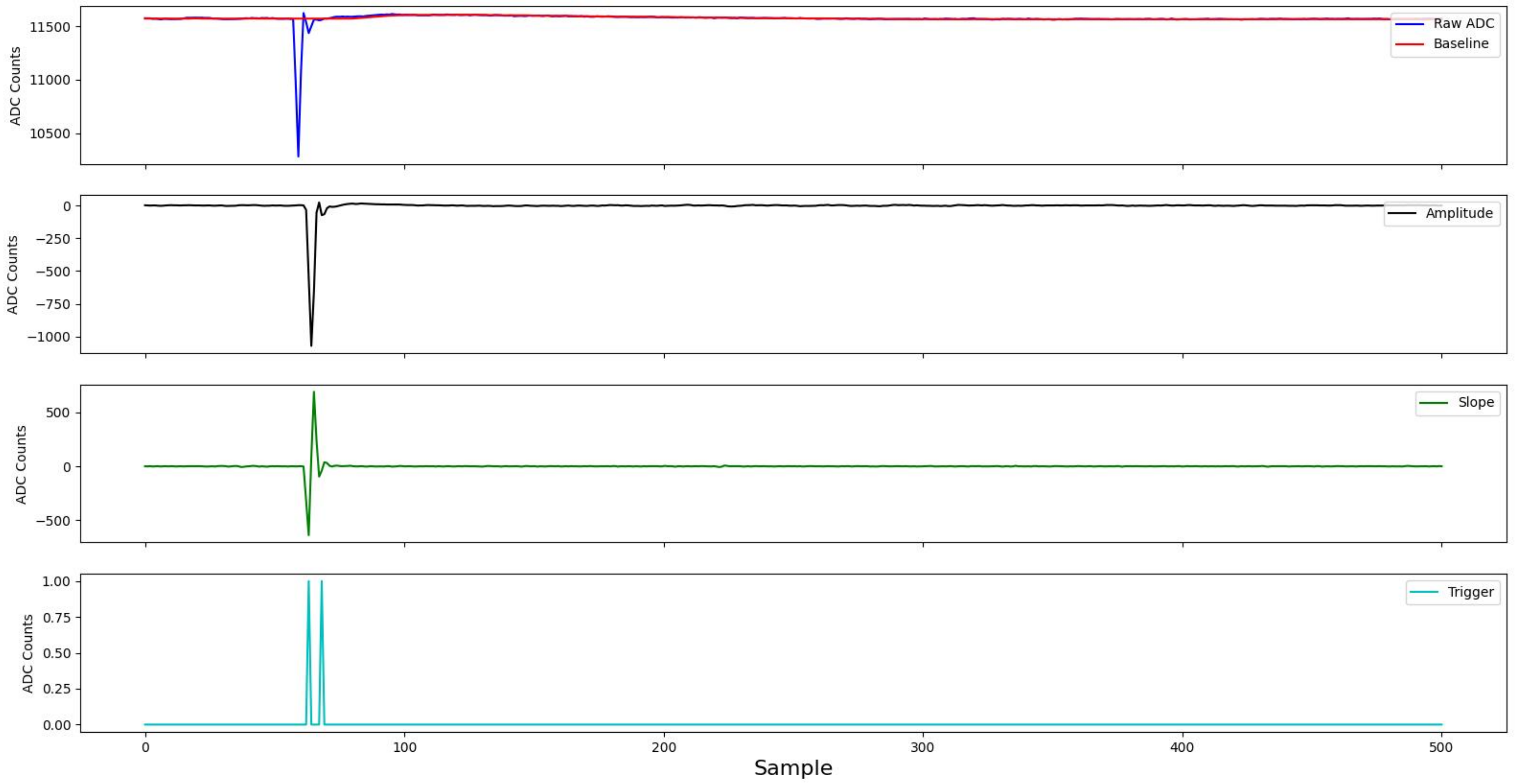
- Design a circuit that meets the required specifications.
- Prototype and test the designed circuit.

Future plans

- Tune the circuit to obtain better response.
- PCB layout design.
- Implement a block to check trigger coincidences between channels.

Adapt Daphne for PMTs data acquisition (4)

Signal acquisition with our Self-Trigger algorithm



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** $NP_{i+1} = NP_i$