Implementation of Time to Digital Converter on FPGA for high-resolution-performance in Particle Therapy applications

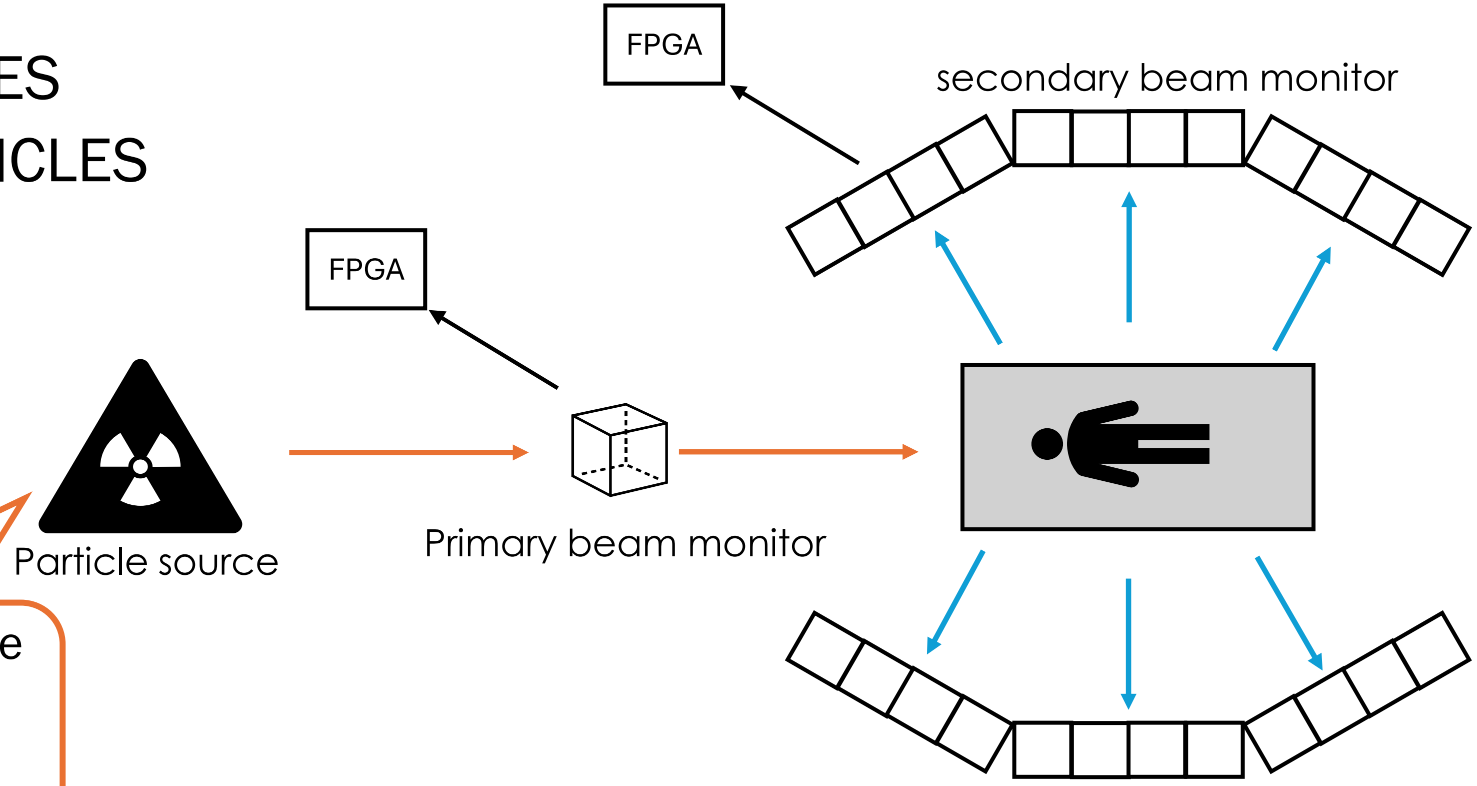
Arash Amini Bardpareh[1], Eleonora Vacca[1], Corrado De Sio[1], Sarah Azimi[1], Luca Sterpone[1]
Elisa Fiorina[2], Emanuele Maria Data[2], Felix Mas Millian[2]
[1]*Dipartimento di Automatica e Informatica, Politecnico di Torino*, Turin, Italy
[2]*Istituto Nazionale di Fisica Nucleare*, Turin, Italy

Introduction

- HONEY: Hybrid Online tEchnologY for particle therapy

Introduction

- Real-time monitoring of beam parameters to fine-tune for patient care
  - PRIMARY PARTICLES
  - SECONDARY PARTICLES
  - DAQ on FPGA



The **goal** is to measure the time difference between primary and secondary particles

Fig 1: Treatment technique during Charged Particle Therapy (CPT)

Background: Time-to-Digital Converter

- Working Principle
  - An electronic device used to measure very short time intervals with high precision by converting time differences into a digital value.

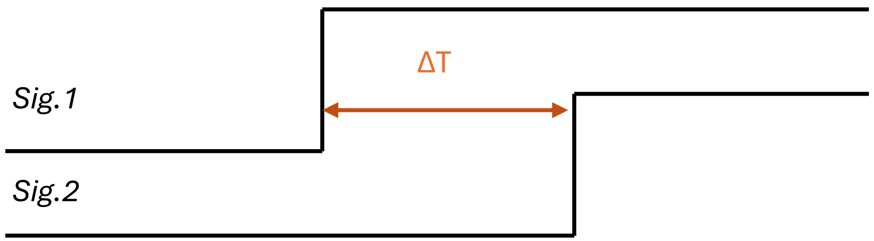


Fig 2: TDC Basic Functionality

State Of The Art Architecture

- Tapped Delay Line
  - Signal propagates through a series of delay elements
  - A register to capture the state
  - Its Advantages
    - Its simplicity
    - High resolution

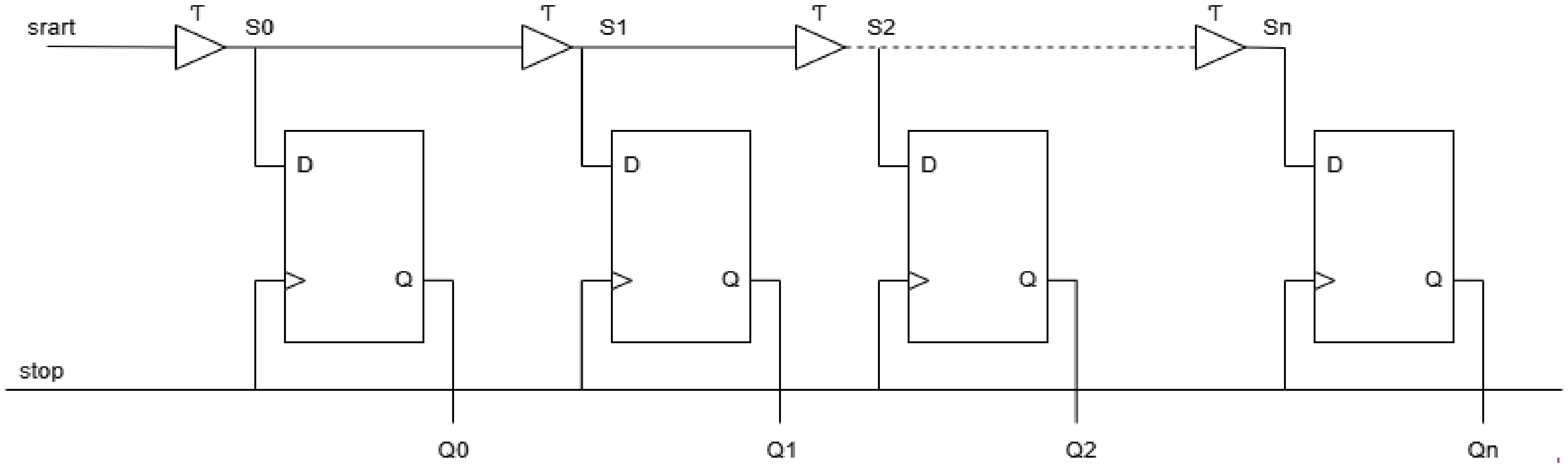


Fig 3: Tapped Delay Line Schematic

FPGA Logic Resources

- In an FPGA, limited Elements can be used to generate the delay
- All we can use is what we have Inside CLBs
  - LUTs
  - Multiplexers
  - Registers
  - Carry Logics

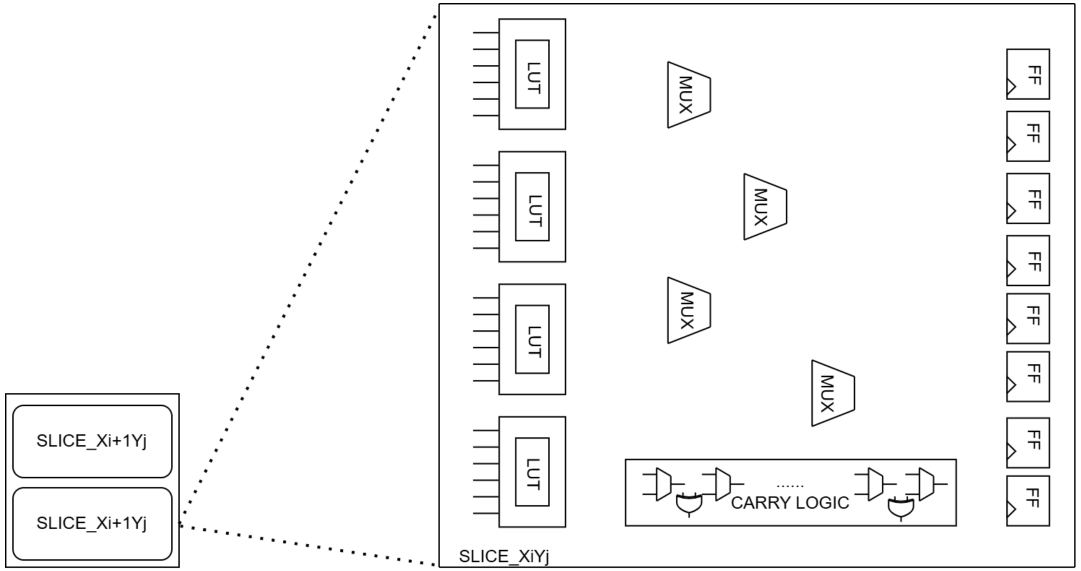


Fig 4: CLB

FPGA- based Implementation

- Carry Blocks
  - **Fast lookahead** carry logic for fast arithmetic add/sub
  - Being able to connect the in **cascade**
  - Register with the **short route**

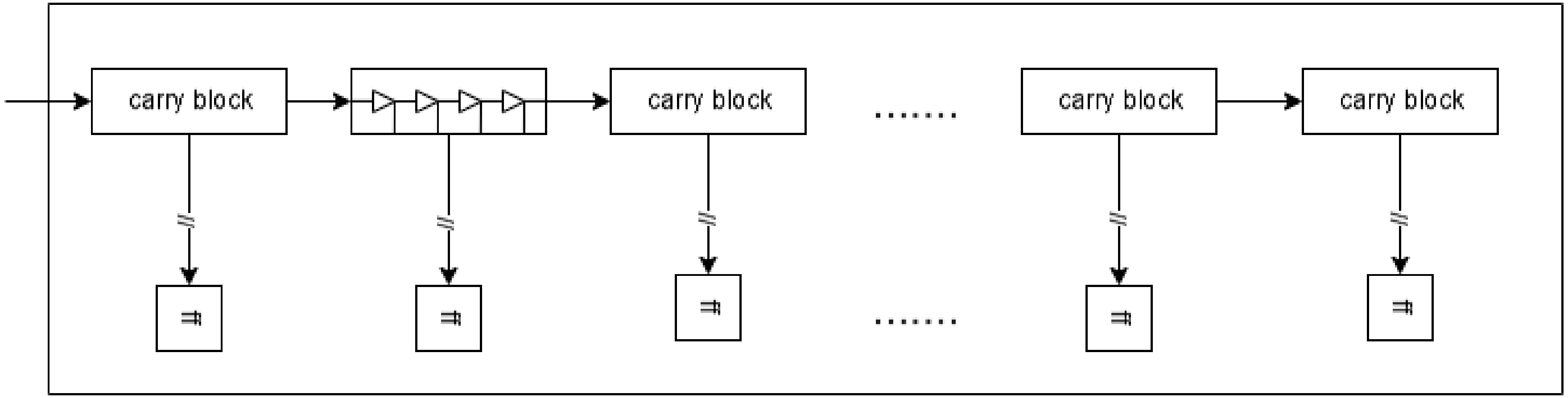


Fig 5: Implementation of the delay line in FPGA Using Carry blocks

Timing Simulation After Implementation

- Zoomed View of Delay Line
  - The carry chain connections
- Simulation doesn't show delay inside the carry blocks
  - Go to experimental test

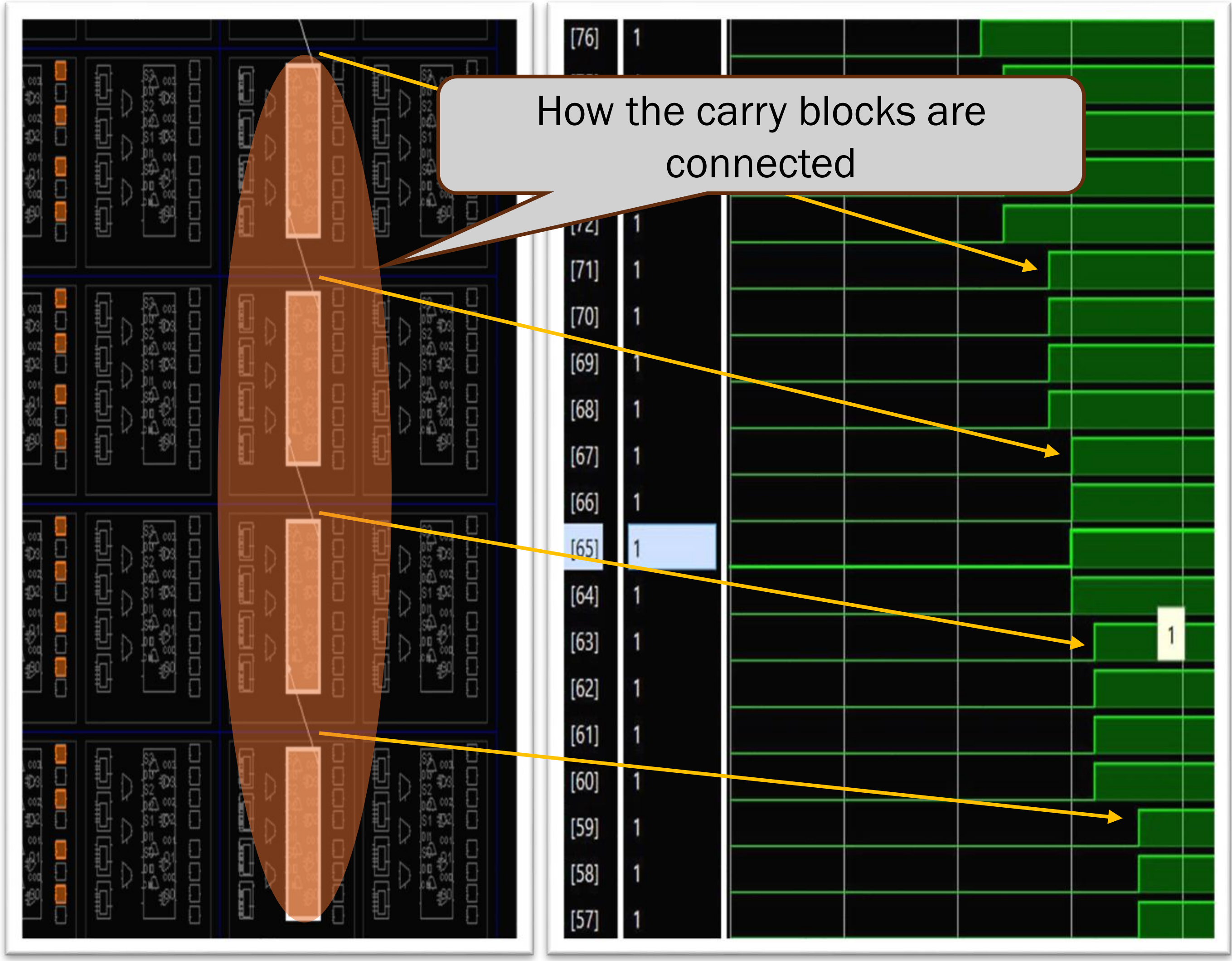


Fig 6: Part of Delay Line output in vivado simulation

Timing Simulation After Implementation

- Zoomed View of Delay Line
  - The carry chain connections
- Simulation doesn't show delay inside the carry blocks
  - Go to experimental test

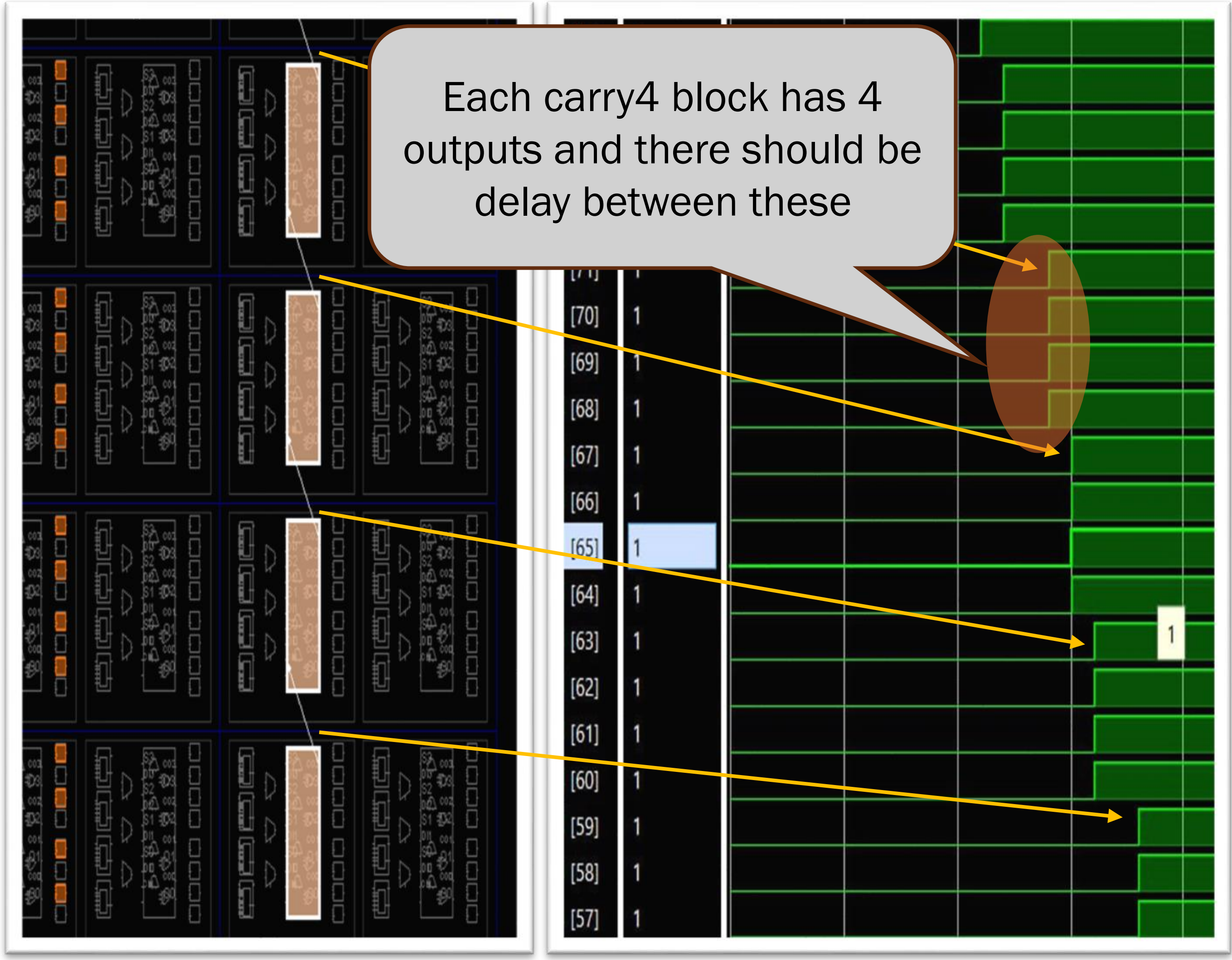


Fig 6: Part of Delay Line output in vivado simulation

Timing Simulation After Implementation

- Zoomed View of Delay Line
  - The carry chain connections
- Simulation doesn't show delay inside the carry blocks
  - Go to experimental test

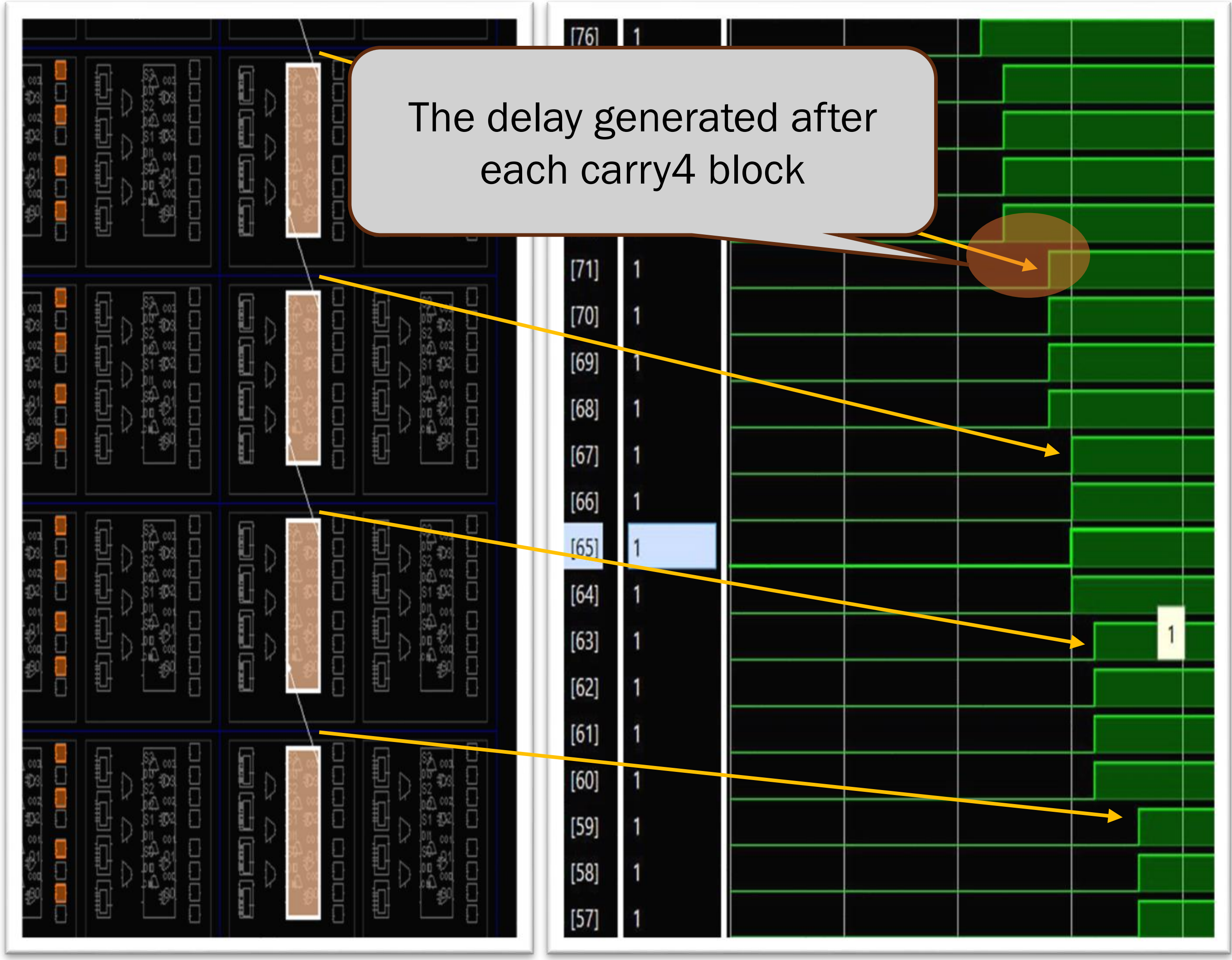


Fig 6: Part of Delay Line output in vivado simulation

Expected Behavior

- Logic delay is τ
  - "1110" shows signal propagated through 3 delay elements
  - Delay : 3 x τ ps
- Registers become high linearly

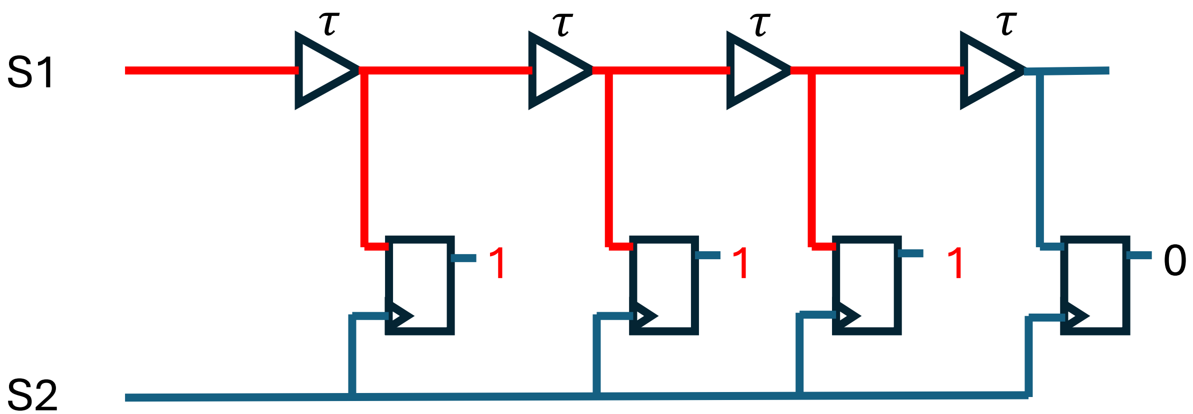


Fig 7: signal propagation in delay line - expected

Real Behavior – Bubble Errors

- Logic delay is τ
  - "1010" shows signal propagated through
    - 2 delay elements → Delay = 2 x τ
    - 3 delay elements → Delay = 3 x τ
  - This behavior is known as " bubbles "
    - Due to variation in fabric
    - Small asymmetry in implementation and routing

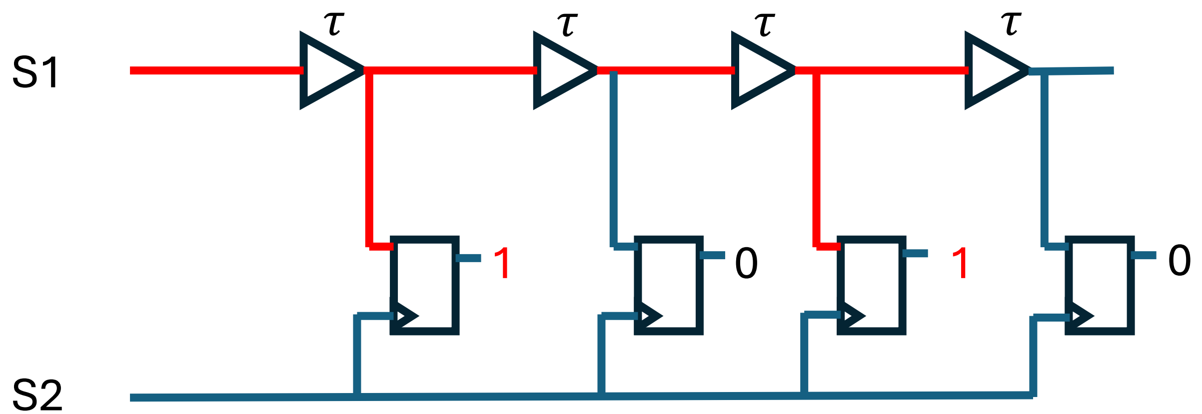


Fig 8: signal propagation in delay line - Real

Increase Measurement Range

- It’s important to keep the delay line on the same clock region to avoid clock skew
  - Max possible length is 464 taps (58 carry8 blocks)
  - Each delay line can measure up to 2.8 ns
- Leads to modifying the design to extend the dynamic range
  - An additional synchronous counter to measure time if it’s more than the system clock period

The proposed TDC Working Principle

- Divide the delay measurement into three parts
  - Time difference between the asynchronous arrival of the first signal (*start*) and the next rising edge of the clock – **by a delay line, fine-grain measurement**
  - Time difference between synchronous first signal arrival (*start*) and the synchronous second signal arrival (*stop*) – **by a counter, coarse grain measurement**
  - Time difference between the asynchronous arrival of the second signal (*stop*) and next rising edge of clock (by delay line)

$$T = T1 + T2 - T3$$

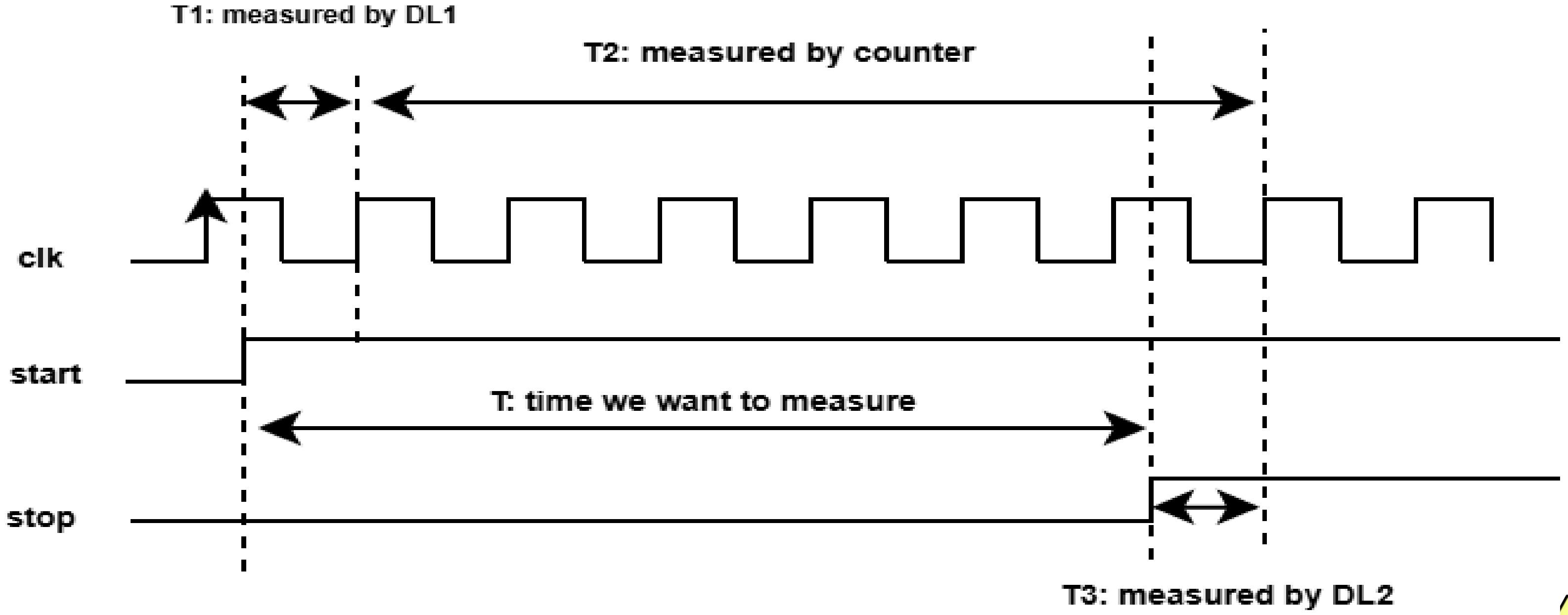


Fig 9: Role of each module for measurement

TDC Block Diagram

- The fully implemented TDC architecture

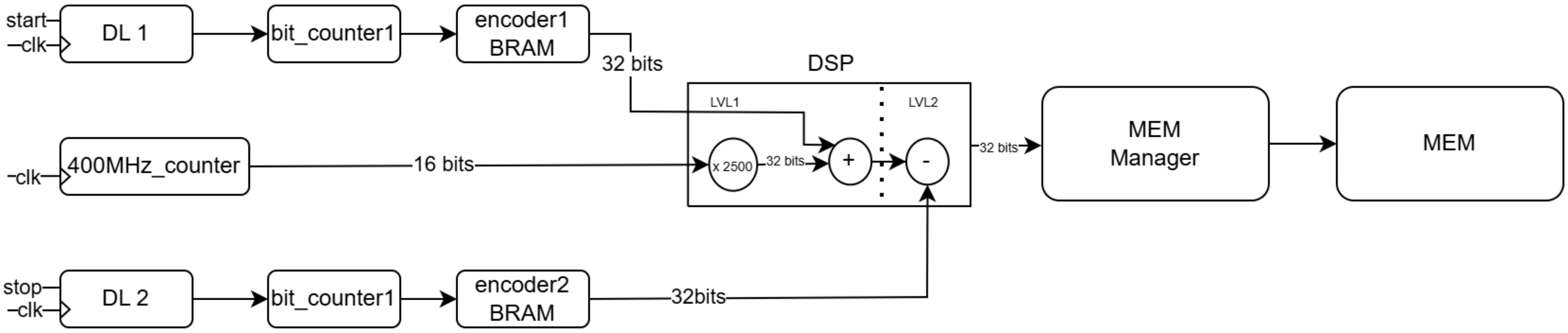


Fig 10: Schematic of final design

TDC Block Diagram

- The fully implemented TDC architecture

- ❑ Receive 464 bits in groups in parallel
- ❑ Count the 1's in multiple stages
- ❑ Using LUTs and fabric adders
- ❑ It matches the system's measurement rate

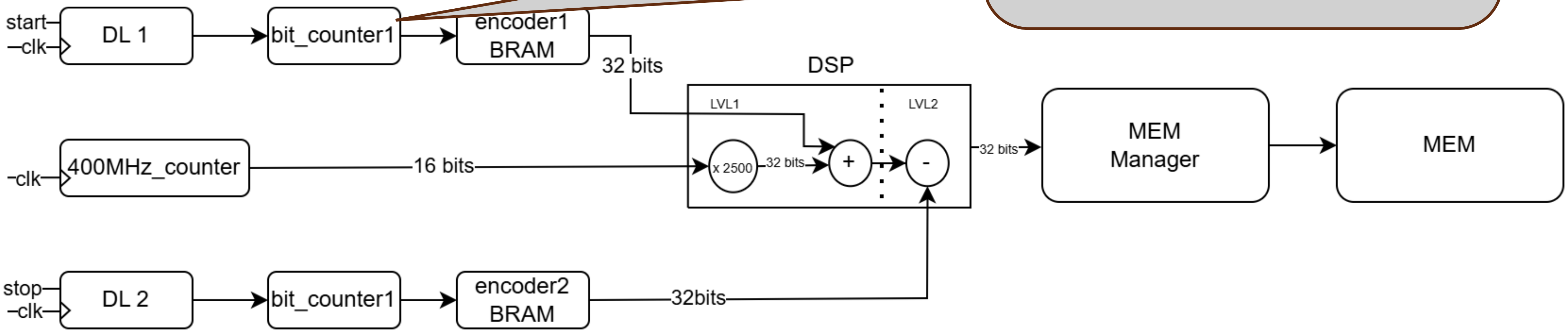


Fig 10: Schematic of final design

TDC Block Diagram

- The fully implemented TDC architecture

- ❑ Memory is implemented to save the results as a histogram
- ❑ Adaptable to generic bin size and measurement range

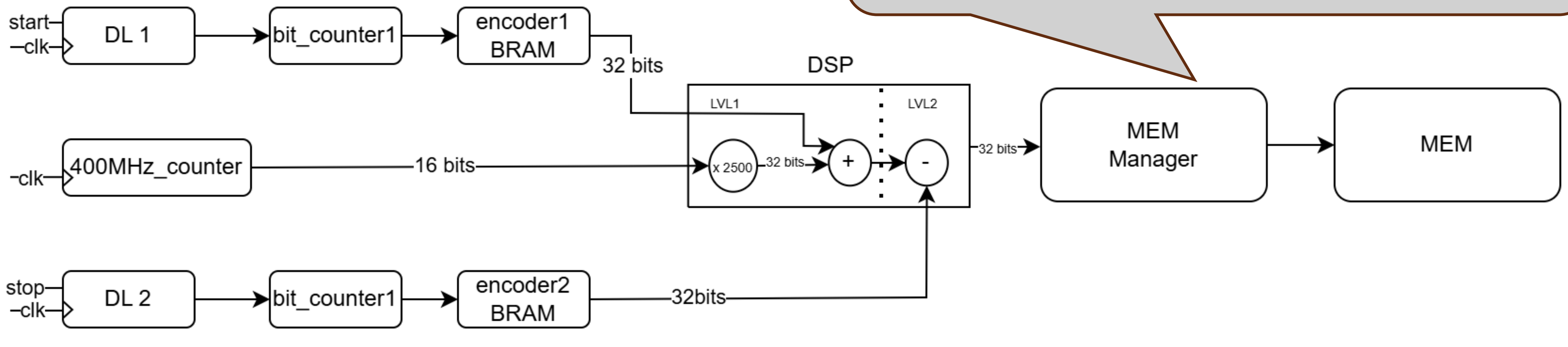


Fig 10: Schematic of final design

TDC integration with ESA-ABACUS

- TDC receives signals from the ESA-ABACUS
- ESA-ABACUS includes:
  - a 50-μm-thick Low Gain Avalanche Diode (LGAD) sensor segmented into 146 strips
  - six 24-channel, each having a charge-sensitive amplifier followed by a leading-edge discriminator
- Since the expected clinical particle rate (and ESA-ABACUS output) is in the range of 1 MHz, the TDC is designed to comply 10 MHz measurement rate
- The input signal width from the ESA-ABACUS is 2ns
  - Use latches to hold the signal to avoid losing data
- Specific placement to avoid asymmetry in the design

Non-Linearity Issues

- Different routes for each DL can cause offset
  - The measurement starts when sig1 enters DL1 and ends when sig2 enters DL2
    - If x2 > x1, it causes an external offset, which can be found and removed
    - If x1 > x2, it causes an offset, which causes a problem when small measurements

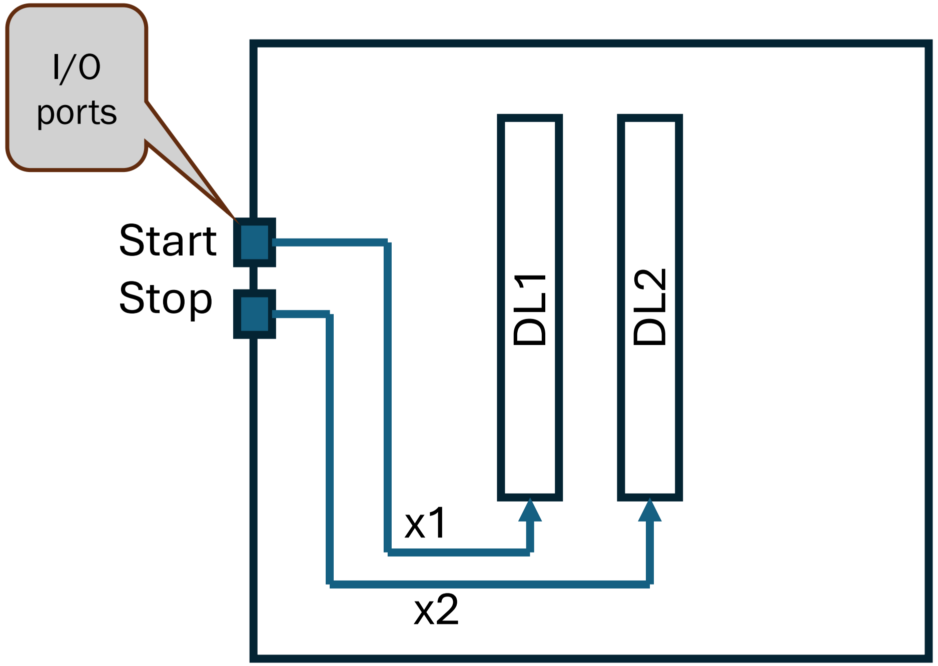


Fig 11: routing from ports to design

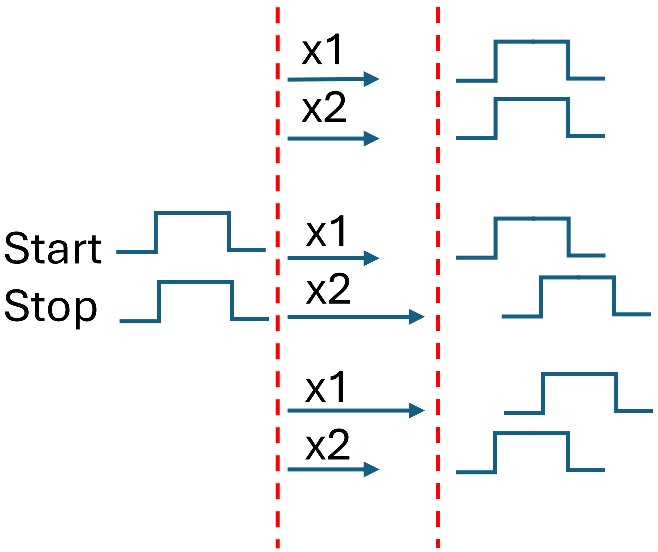


Fig 12: external offset effect caused by routing

Placement Issues

- Sig1 also activates the counter, and Sig2 deactivates it
  - There is a mismatch in timing between
    - Sig1 enters the DL1 before enable the counter
    - Sig2 enters the DL2 before disable the counter
      - Counter will count 1 more /or less

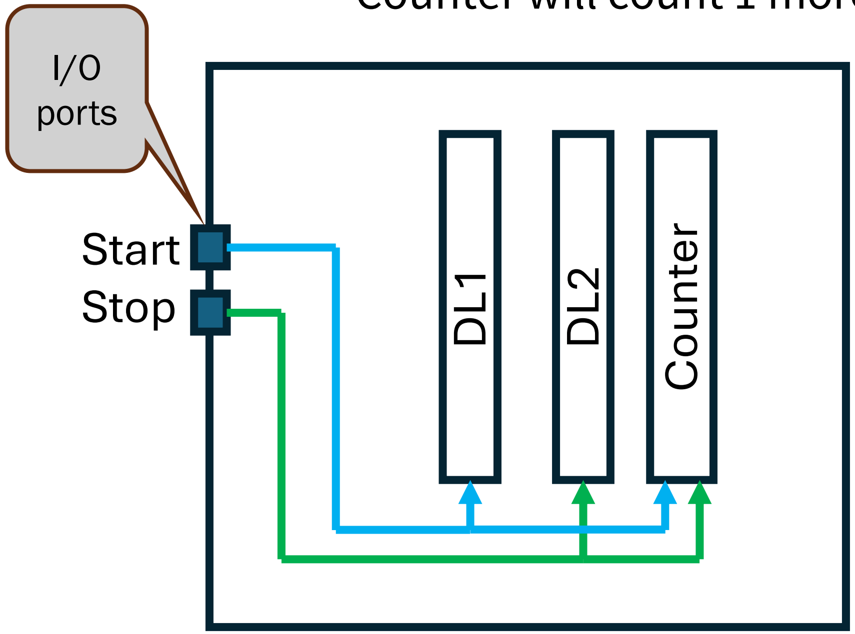


Fig 13: time mismatch between DLs and counter

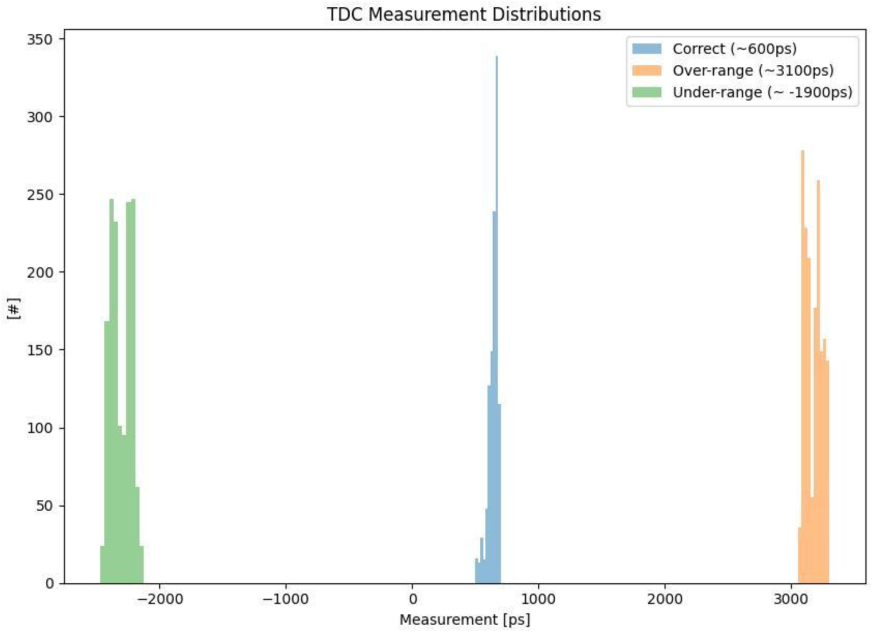


Fig 14: wrong measurements due to timing mismatch

Placement

- Implemented design on Xilinx Kintex Ultrascale KCU105

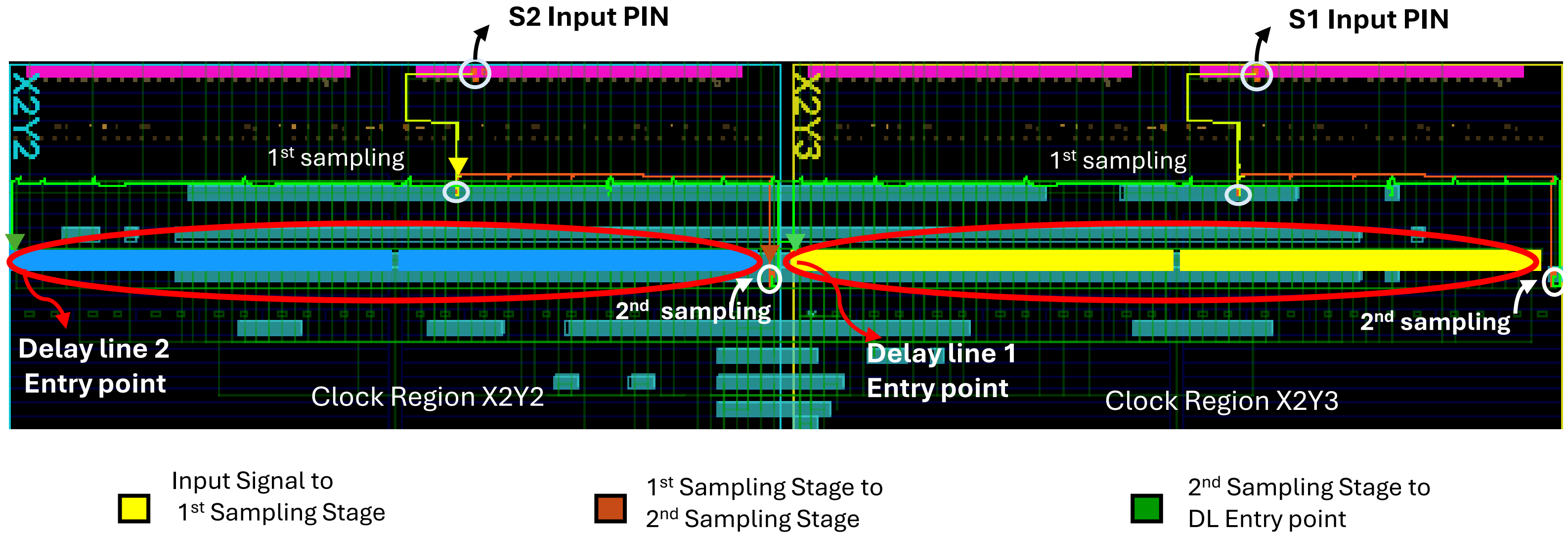


Fig 15: implementation on FPGA

TDC Output

- Gathering data from the delay lines
  - Finding the best calibration from the most repeated output
  - By weighted mean
  - TDC works with max median error of ~25ps and σ varies between 30 to 135 ps.

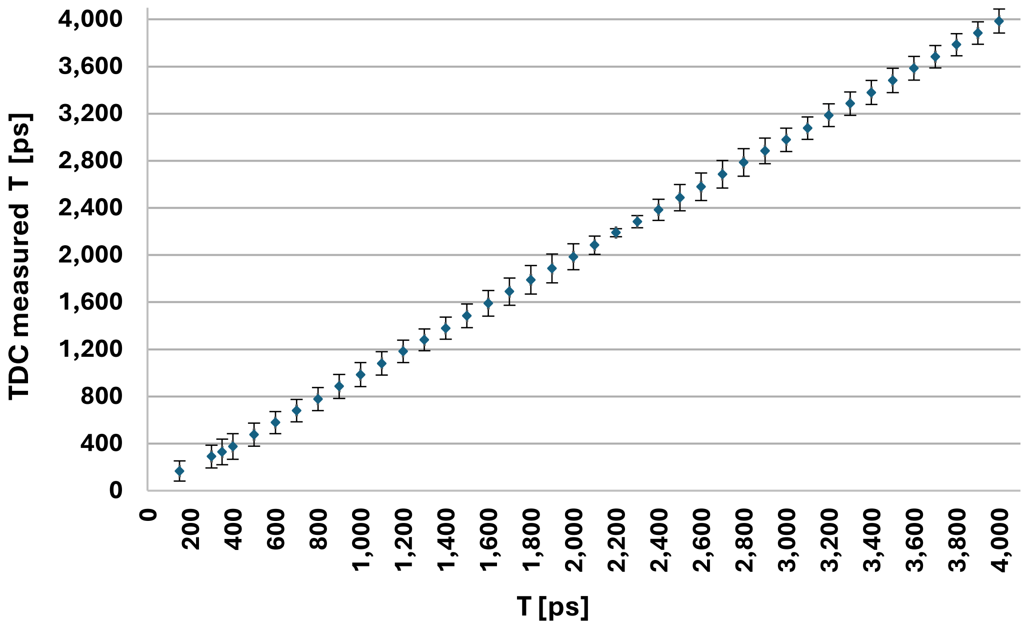


Fig16: TDC input-output characteristics

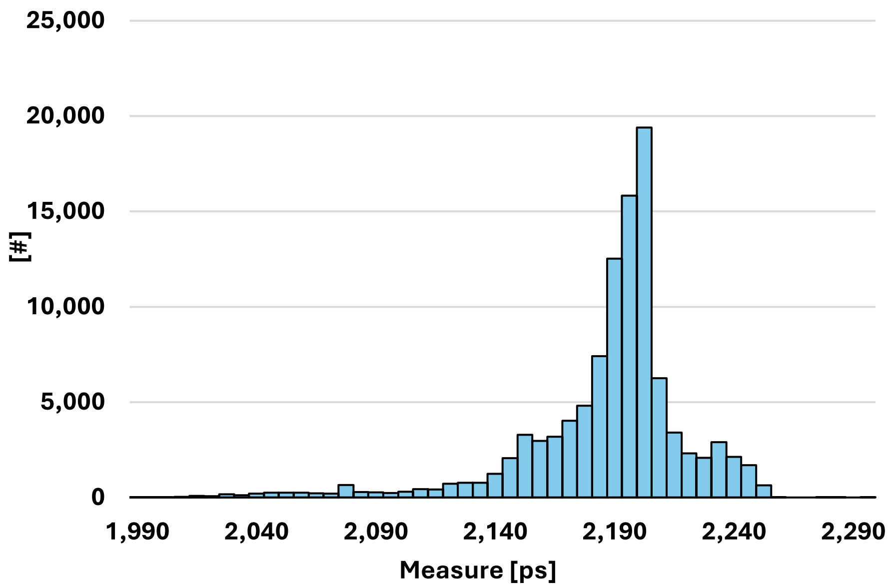


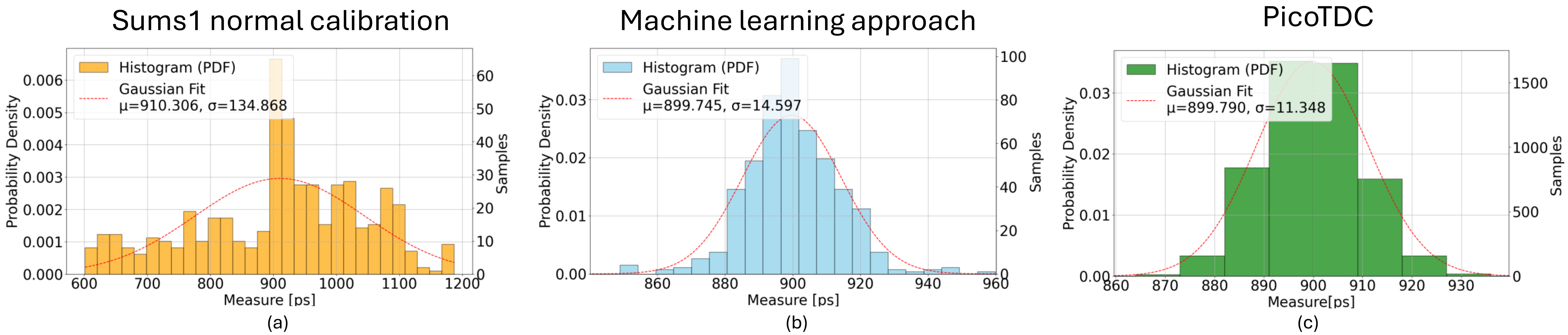
Fig17: Distribution of measurements for 2200ps

Calibration: A Machine Learning Approach

- A MLP model consisting of two layers, with a total of 64 neurons, to improve calibration
- The NN receives a raw stream of bits coming from the two delay lines and the counter
- Bit counter and encoder are removed from the design

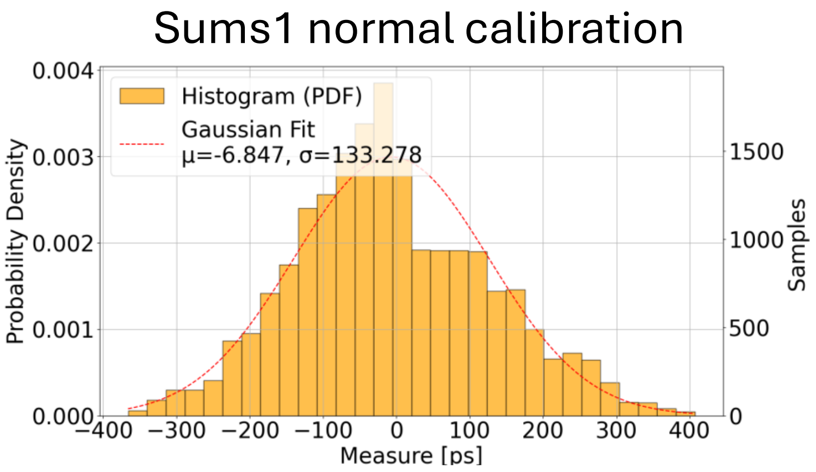
Example of 900ps measurement

- The σ improved from ~135 ps to 14.6 ps, achieving quality comparable to the ASIC picoTDC

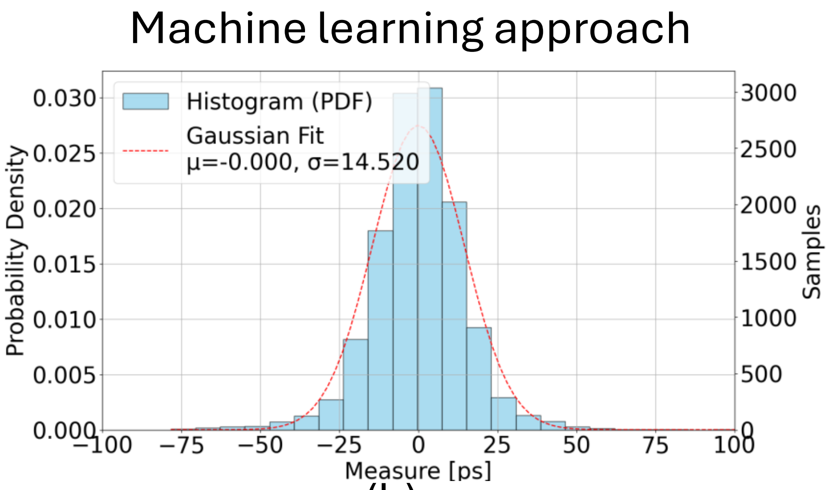


Error distribution

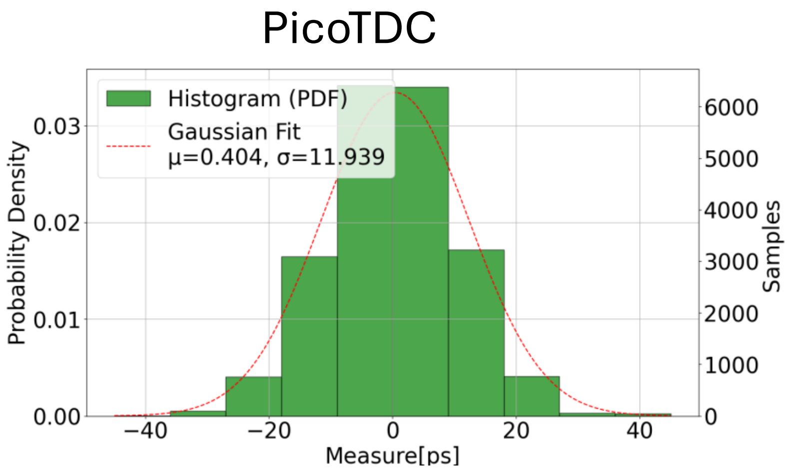
- The improvement in the distribution of error with the ML approach



(a)



(b)



(c)

Conclusions and Future Work

- Calibrating with NN brings excellent results
- However, offline processing is slow
- Integrating a Systolic Array accelerator along the TDC, to speed up the NN inference execution
- Utilizing the dual sampling in the tapped delay line method to further increase the resolution

Thank you for your attention