Integration and optimization study of a large-scale logic circuit for the first-level muon trigger at HL-LHC

TDAQ Week in Kyoto, 18-22 Sep. 2023
Yoshifumi Narukawa(ICEPP, The University of Tokyo)

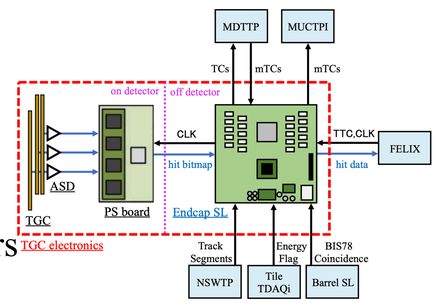
1.Abstract

For the high-luminosity LHC-ATLAS experiment, the development of the first stage muon trigger logic has been underway. In this research, I will focus on two main topics. The first topic is the performance evaluation of the trigger logic using the actual hardware. I successfully constructed a trigger calculation system using real machinery, which enables the emulation of trigger responses to physical events generated through Monte Carlo simulations. The second topic is logic optimization aimed at the integration of all trigger logics. I managed to significantly improve timing constraints by altering the implementation method of the shift register and optimizing the track selection logic.

2.L0 muon trigger system

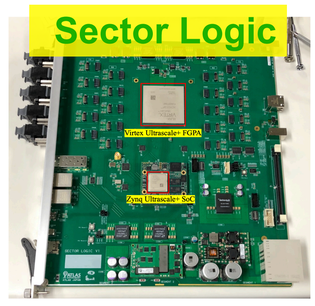
Phase-2 TGC electronics overview

- Front-end electronics (1434 board)
  - Sends all hit bitmap to Endcap SL with fixed latency (8Gbps x 2)
- Endcap SL (48 boards)
  - Performs muon track reconstruction and estimates Pt
  - Improves Pt resolution using information from inner muon detectors
  - Readout hit data according to L0 accept to FELIX (9.6 Gbps x 3)



Endcap SL hardware overview

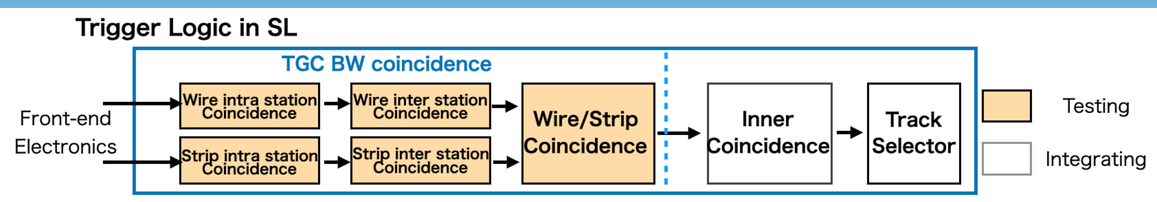
- Virtex Ultrascale+ FPGA
- Zynq Ultrascale+ MPSoC
- IPMC
- FireFly TX (12ch) x 10
- Fire Fly RX(12 ch)x 10
- Clock jitter cleaners



3.Trigger firmware status

- Trigger Logic is constituted by five-stage coincidence logic
  - The first three stages are called TGC BW coincidence (use only TGC hits)
  - Inner coincidence utilize information from inner muon detectors
- Implementation of TGC BW coincidence has been completed
  - Performance evaluation using software based simulation
  - Firmware implementation

▼The next important step is to validate trigger performance using actual SL



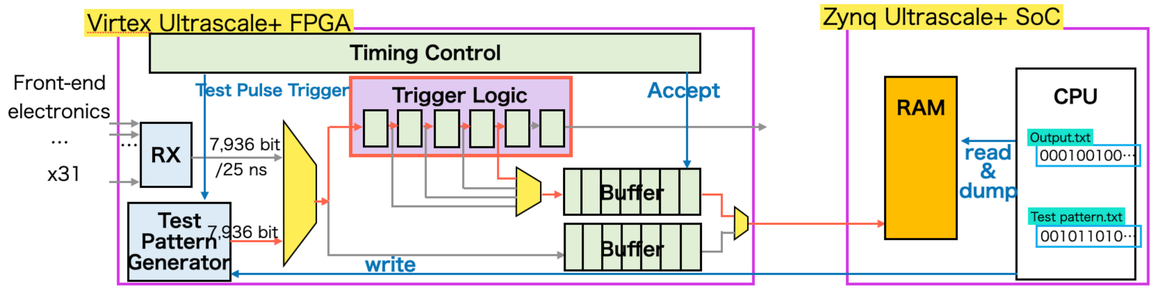
- Integration of inner coin is ongoing
  - At present, congestion occurs, leading to timing violations

▼Exploring ways to resolve timing violations without sacrificing performance

4.Trigger performance test using SL

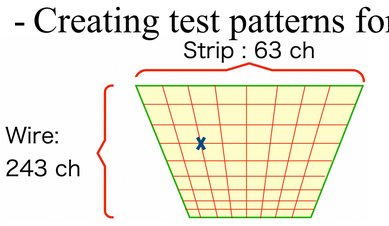
Overview of the trigger logic test system

- Test patterns are used as the input of the firmware
  - The Test Pattern Generator embeds a 7,936 bit hit data that emulates the 31 front-end electronics in a certain BC
  - Test patterns are stored in BRAMs and can be overwritten again
- TTC is emulated in the FPGA
  - Trigger logic works with fixed latency
  - Synchronize Test Pulse Trigger and L0A to acquire only the trigger output for test patterns at a certain BC
- Able to switch which trigger module's output to acquire

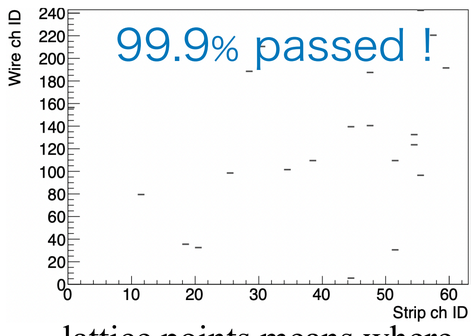


Validation of the test system using infinite momentum track

- Create test pattern that emulate infinite momentum track
  - Muons entering the TGC linearly from the IP
  - Track should be 100% reconstructed with TGC BW coincidence
  - Creating test patterns for all lattice points



Wire 243 ch
x Strip 63 ch
= 15,309 pattern

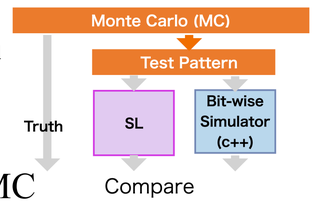


- 15,288 / 15,309 pattern passed
  - Test system running with high accuracy
- Irreducible errors (~0.01%) of either on the readout or the trigger logic exits

lattice points means where track reconstruction failed

Firmware validation injecting single muon event data

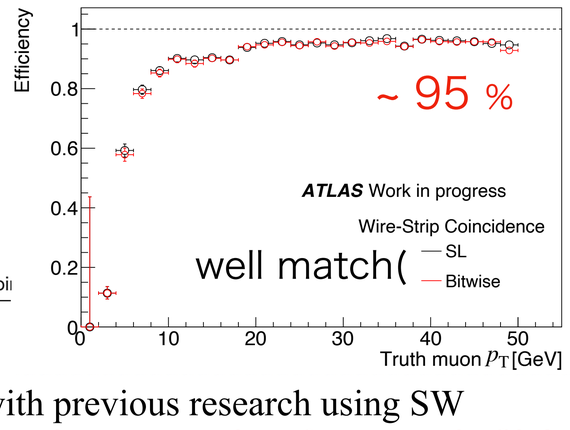
- Realize hardware testing using input patterns derived from simulation data
  - Utilizing MC simulation data (single muon)
  - Creating test pattern from the detector's hit simulation
  - A bitwise simulator emulates trigger logic in C++
  - Compare outputs from SL, Bitwise, MC truth



- Validating reconstruction efficiency using Single Muon MC

Event	Single Muon/ 0< p_T < 50 GeV / no pileup / 29K event
Cut	Excluding the chamber boundaries (1.9 < eta < 2.0, the outer 10% in the phi direction)

$$\text{efficiency} = \frac{\text{The number of muons reconstructed by TGC BW coi}}{\text{The number of Truth muon that passed cut}}$$

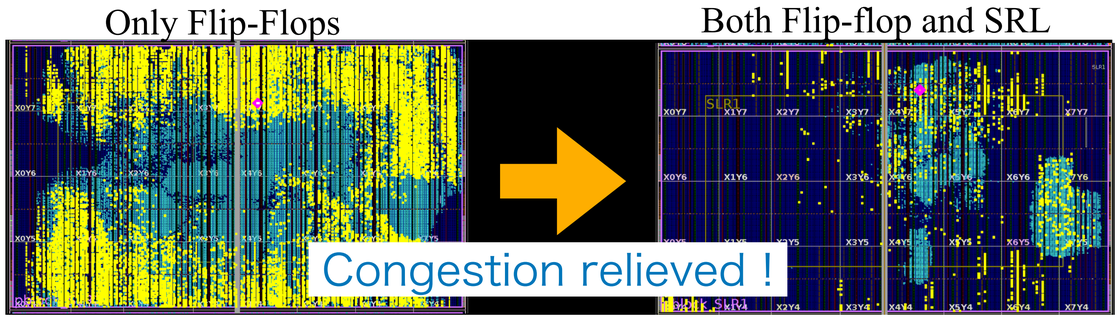


- The trigger efficiency matches well with previous research using SW
  =>I can verify that the trigger logic has the expected performance in SL !

5.Optimization to resolve timing violation

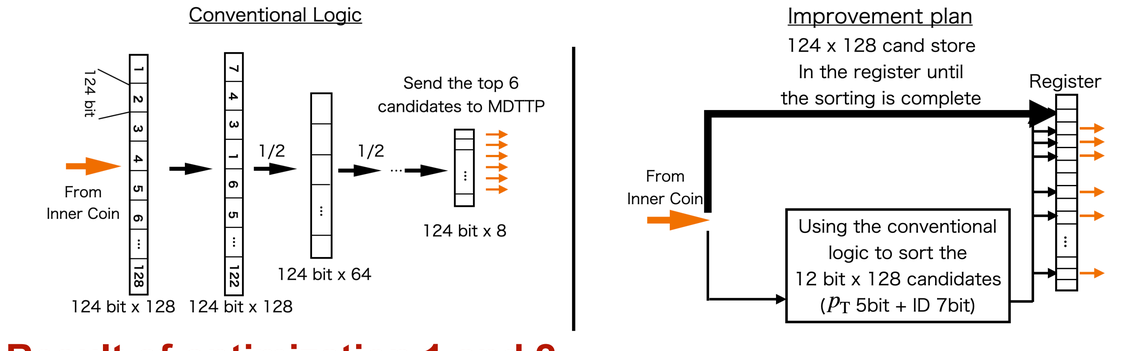
1.Optimization of pipeline register implementation methods

- The common approach to relaxing timing constraints is to insert shift registers between logics where timing violations occur
  - The shift register acts as a relay point in signal propagation
  - The flexibility of primitive placement increases
- Two different method for implementing shift register in Ultrascale+ FPGA
  1.Flip-flop (FDRE)
    - Basic primitives included in a typical CLB / hold only 1bit
  2.Shift Register LUT (SRL)
    - Primitive specialized for the shift register /hold 1bit x 16 depth
- Shift registers have been implemented using only flip-flops; I have made changes to also utilize SRLs
- Registers that do not function as relay points are now stored in SRLs, improving density and, as a result, significantly reducing timing violations!



2.Optimization of track selection logic

- The final stage of the trigger is sorting logic that selects six out of a 128 muon track candidates to send to MUCTPI
- By eliminating the waste in the sorting logic, situations where more than 10,000 bits of data move between flip-flops has been resolved



3.Result of optimization 1 and 2

- TNS(Total Negative Slack) has been significantly improved !
  - However, timing violation still remain => further optimization is in progress

TNS : -197,735 ns → -45,316 ns → -0.2 ns