Track-Muon Trigger Correlator Demonstrator Status

Sven Dildick, Jason Gilmore
Evaldas Juska, Alexei Safonov

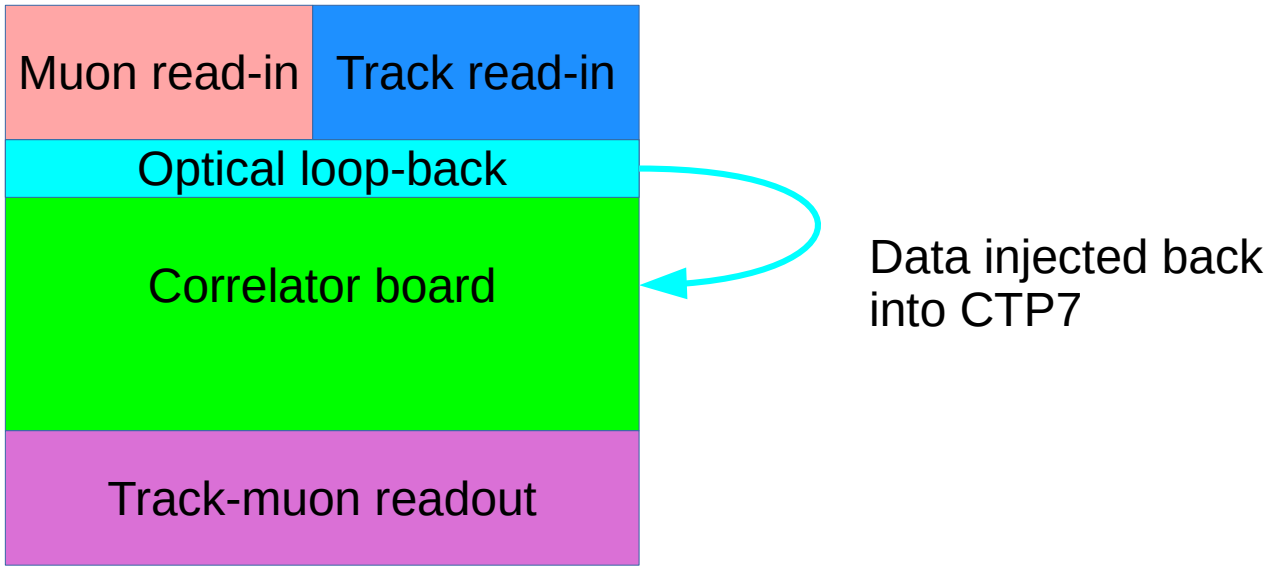
Texas A&M University
November 28, 2018

The Algorithm

- L1 tracks are propagated from the interaction point to the 2nd muon station
  - Algorithm was designed in simulation in the context of the Phase-2 TP (2014) and has good performance (low rate, high efficiency)
  - https://github.com/cms-sw/cmssw/blob/CMSSW_6_2_X_SLHC/SLHCUpgradeSimulations/L1TrackTrigger/plugins/L1TkMuonFromExtendedProducer.cc
- Propagated tracks are matched to the muons, based on proximity in eta-phi spac
  - For the time being, we use a fixed dR cut <0.2
    - Can be improved in the future to a pT-dependent dR cut
- Matches are sorted according to match quality
  - Currently: agreement between track and muon pT
- Best matches are sent out to the next processing step as prompt muons
  - Muons without track match are candidates for displaced muon
- Algorithms works for tracks and muons in **barrel, overlap and endcap region** (eta<2.4) and for **0<phi<2pi**

From simulation to hardware...

- We are currently building a prototype for the track-muon trigger within the correlator framework
- We want to assess implementability and evaluate resources to the point where it would be useful to quantitatively design the system
- Algorithms are implemented in firmware using Xilinx tools and run on actual CTP7 hardware
  - Simulated data is injected through software and delivered over physical fiberlinks
- At this time we are testing up to ¼ of a realistic sector

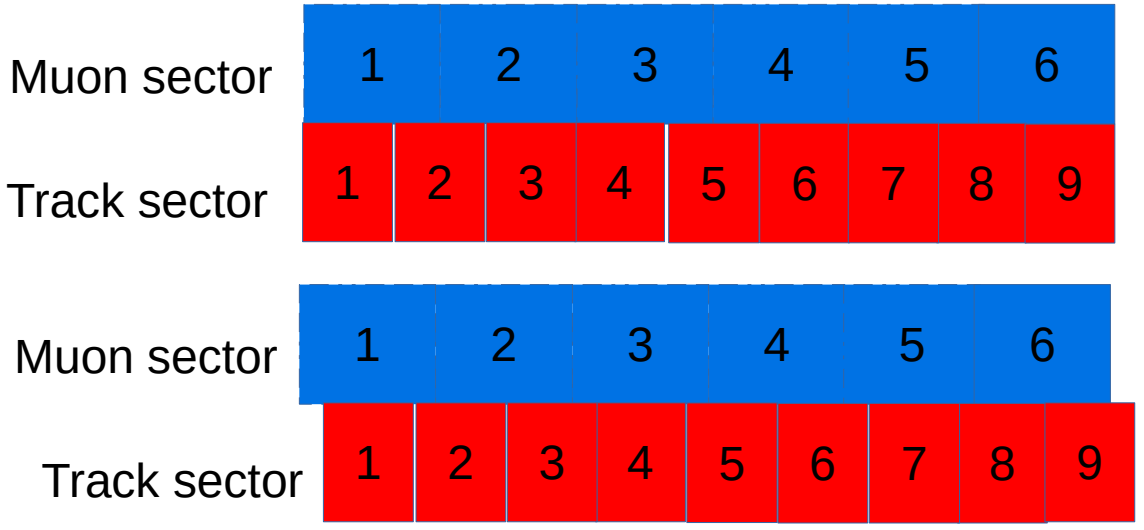


Assumptions and External Inputs

- Throughout these slides we make the following assumptions (based on interim trigger TDR 2017):
  - Trigger correlator uses divided into 9 track sectors
  - Muons trigger uses 6 sectors
  - ~900 tracks for full system –> 100 tracks per track sector
  - 72 muons for full system –> 8 muons per track sector
  - Muon and track data format uses about 100 bits
    - We actually use 96 bits for each object
  - Because the muon and track dataformats are not fully defined yet, we made up a custom dataformat for each with 96 bits
  - Muons are based on CMSSW simulation
  - Tracks are simulated with the VM FPGA code

Hardware implementation of track-muon correlator

- A **fully working hardware prototype** was designed in VHDL
- Core modules written in C++, synthesized with Vivado HLS
  - Propagator module propagates tracks to 2nd muon station
  - Matcher+sorter matches muons to tracks, sorts matches based on match quality
- Correlator designed to cover a **single trigger sector** as for real system
  - About 100 tracks/sector; about 8 muons per sector (24 muons including overlap)
  - Can simplify for the time being: e.g. 1/3 or 1/4 of sector
  - Possible cases under consideration (overlap in phi)
    - Need to consider possible sector overlap cases (+/- 1 muon sector)
- **NOTE: our correlator design is easily translated to a Time Multiplexed system, and this would avoid the complication of dealing with overlaps**



Hardware implementation of track-muon correlator

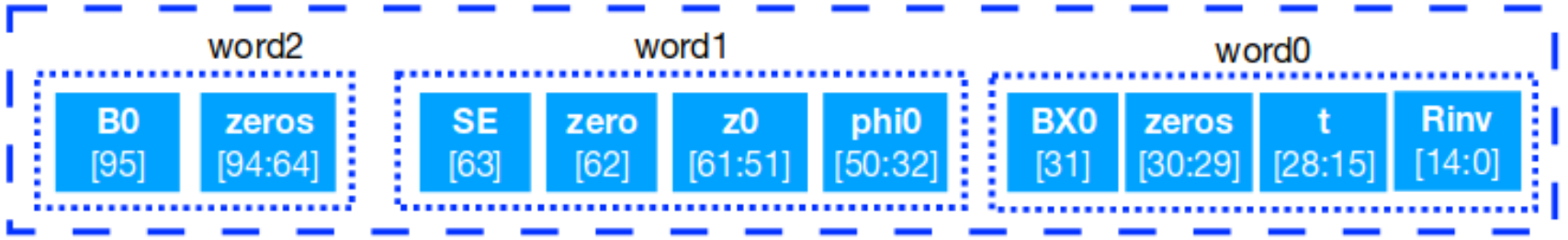
- Can handle a **continuous data stream**
  - New event every BX (internal logic runs @240MHz)
  - Data is received from a single trigger sector (including muon overlap)
- Prototype designed with **necessary flexibility**
- Track/muon datawords (bit assignment) can be changed
- Number of tracks/muons per sector can be changed
- Depends on size of data formats & number of fiber links
- **Realistic firmware** being tested that covers 1/4 of sector
  - With sector overlap we consider 6 muons & 24 tracks
    - 24 tracks works better in our demonstrator than 25 tracks
  - Already being tested on CTP7 (Virtex-7 690T)
  - Verifying/validating the resources for hardware demonstrator
- Full trigger sector coverage to be developed for an ultra-scale device
- Simulation with Vivado suite

Data Formats

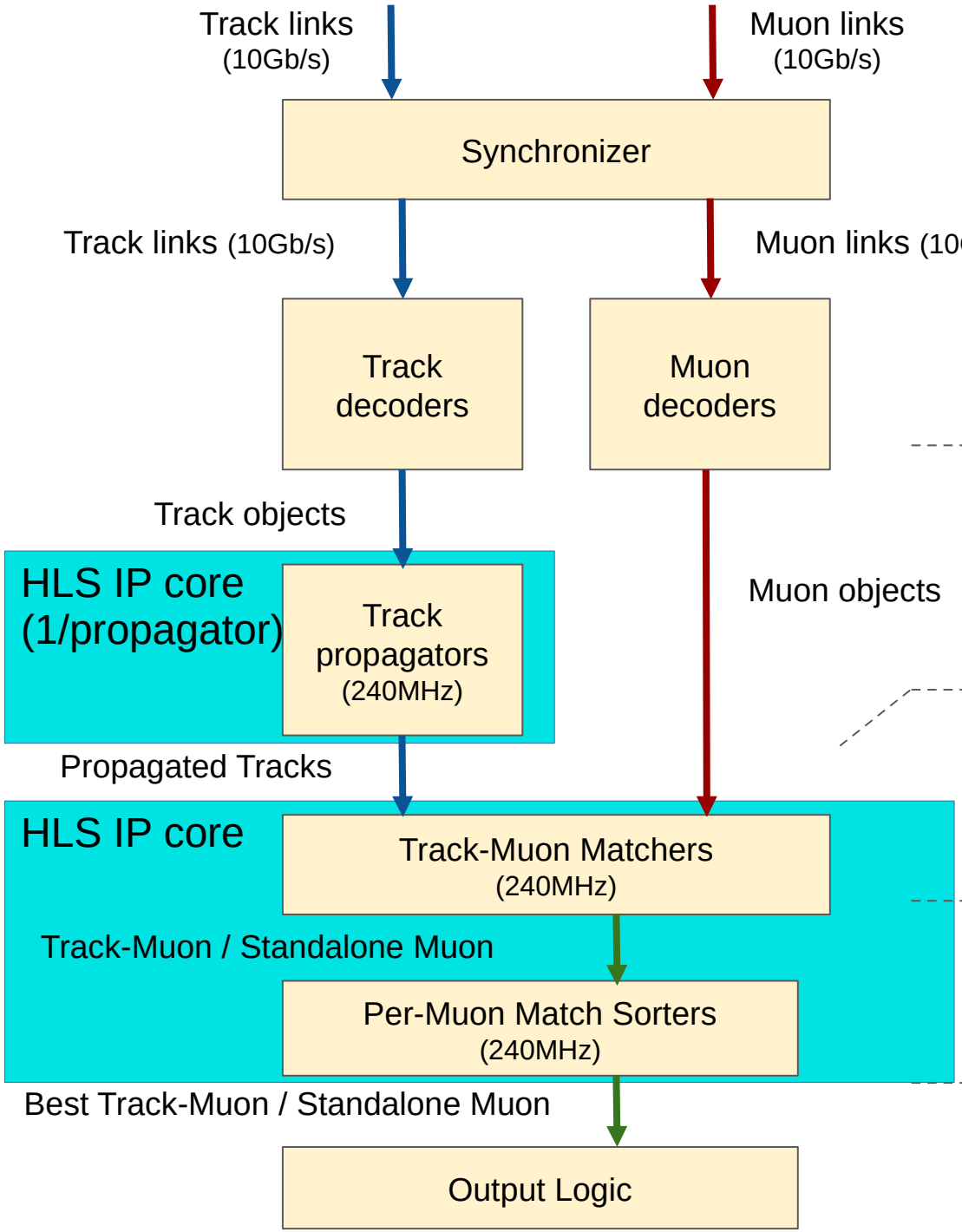
- From Trigger Interim TDR: about ~100 bits per track/muon
- With 10 Gbit/s fiberlinks
  - Can send up to 192 bits per BX per link
- Track and muon, each with 96-bit data word works well in this scheme
  - Can send up to 2 muons or tracks per link per BX
- On ultrascale devices, 16 or 25 Gbits/s --> more bits per object AND/OR objects per link
- In our system, the object data is broken into three 32-bit frames as shown below and this explains the word 0,1,2 diagrams

Muon

Track



Track-Muon Correlator Pipeline Demo with current CTP7



Evaldas Juska (TAMU)

Synchronization of input track and muon data

Decoding of Tracks and Muons
Output:
6 track objects per BX
6 muon objects per BX
(assumes 96-bit track/muon dataformat)

Track propagation to Muon station 2
Operates at 240MHz
One module propagates 6 tracks per BX
Need 17 modules for 100 tracks

Spatial matching of prop. track and muon objects (6 combinations per BX)
Need 134 modules for 8mu 100track
Output: muon [+track and match quality]

Sorting of tracks for each muon based on matching quality
Output only best match or standalone muon

Encoding of the track-muon objects
Displaced muon and veto handling

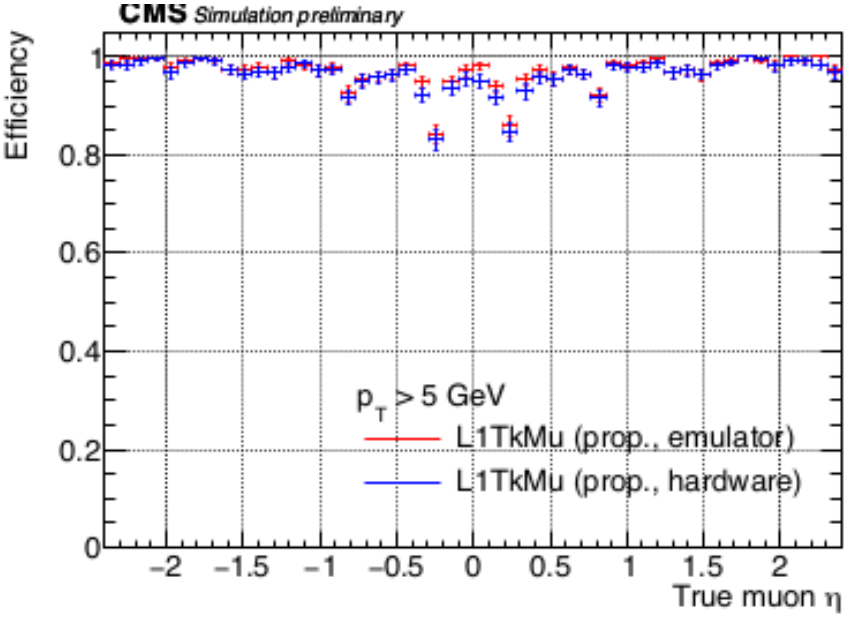
Hardware setup details

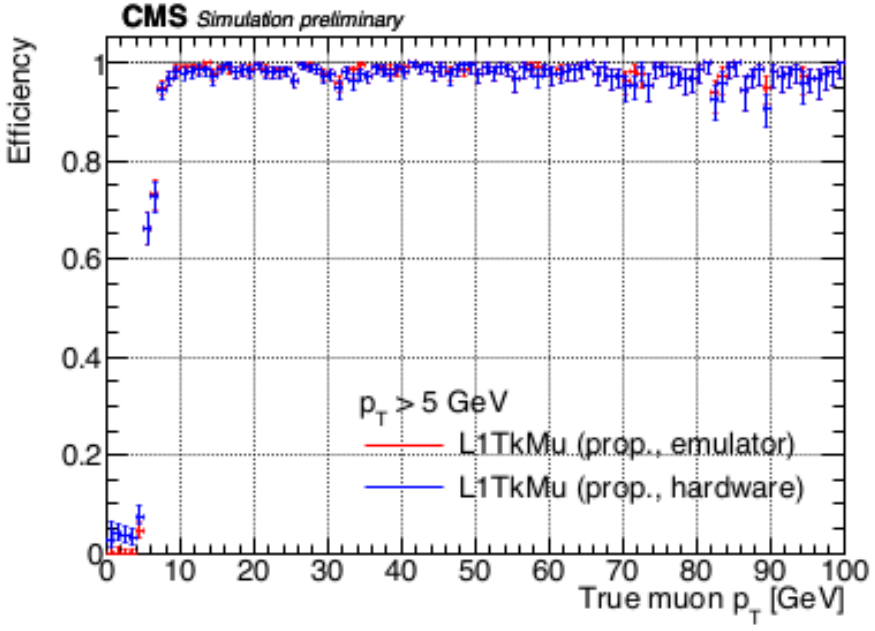
- Test-bench to develop algorithms and test their performance
- CTP7 board with Virtex-7 690T FPGA
- AMC13 provides a 40MHz TTC reference clock and TTC commands
- Correlator logic in CTP7 is using 240MHz clock, derived from the TTC clock
  - Allows to reuse the same logic for 6 objects per BX
- Data is transmitted at 10 Gbit/s using MP7 format
  - Format may change in future
- MTP fiber cables used for track and muon data transmission (loopback)
- Per bunch crossing, each link can transmit 6x32 bits (192 bits)
  - 2 tracks or 2 muons per link (with 96 bits per object)
- External Communication/Control for CTP7 is done through IPBus protocol
- Track propagation & matching module is synthesized as a separate IP core and included into the project

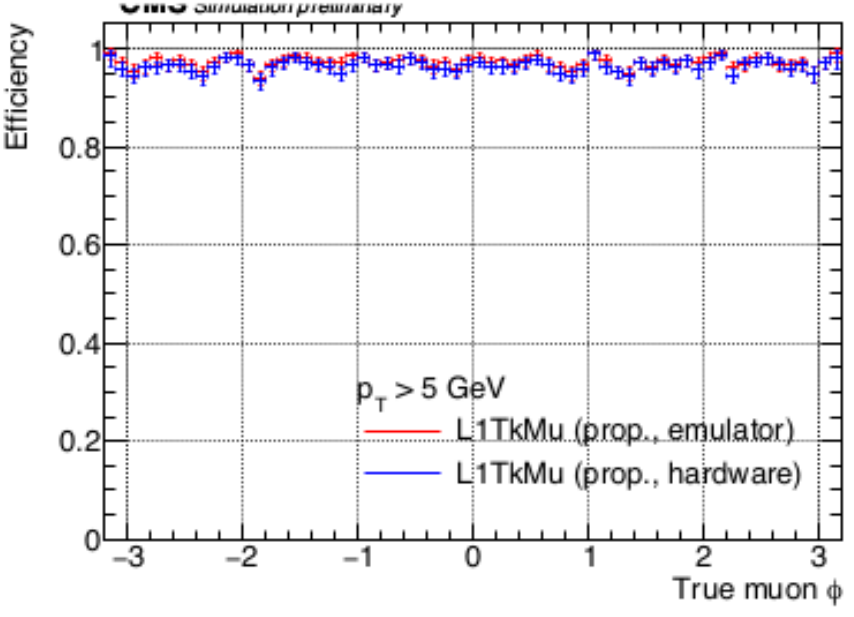
Track-muon correlator trigger performance

- Efficiency to match a simulated muon to a track+muon (emulator vs hardware)
- Good agreement between emulator and firmware

Results for tests with 1 muon link+1 track link







Resource Usage

Resource usage estimates from Vivado HLS for current algorithms (CTP7)

These numbers are **estimates** after synthesizing the C++ code with Vivado HLS **(Not compiled in actual hardware)**

	LUT	FF	BRAM (36Kb)	DSP
Single propagator	4078 (0.94%)	3692 (0.4%)	45 (3%)	37 (1.03%)
¼ sector propagators	16312 (3.76%)	14768 (1.7%)	180 (12%)	148 (4.11%)
¼ sector matchers	308016 (71.1%)	250787 (30%)	9 (0.6%)	432 (12%)
¼ sector total algo	324328 (74.9%)	265555 (31.7%)	189 (12.6%)	580 (16.11%)

¼ sector includes 4 propagator modules (each propagating 6 tracks per BX)
Matcher module handles 6 muons * 24 tracks = 144 objects
Matcher module resource utilization scales linearly with the number of objects = N_muons * N_tracks

Resource Usage

Resource usage in current demonstrator hardware (CTP7)

- These numbers are obtained after compiling the firmware with Vivado
- Firmware Percentages given w.r.t. CTP7 FPGA (Virtex7 690T)

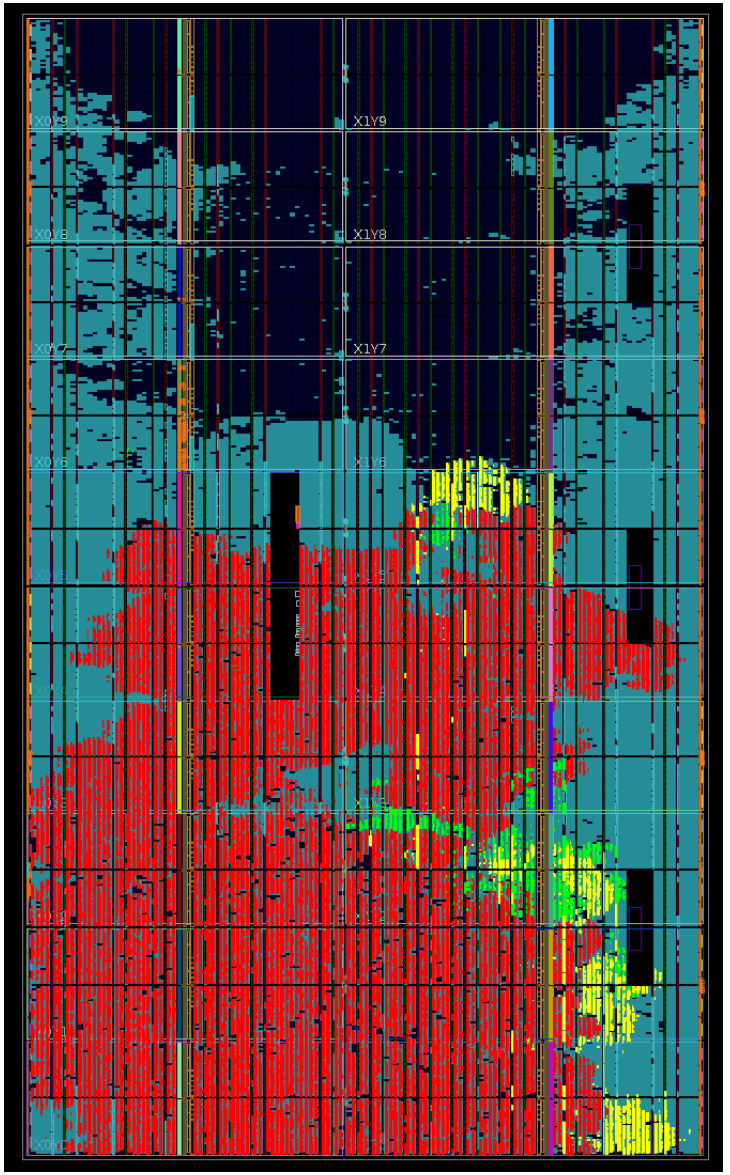
	LUT	FF	BRAM (36Kb)	DSP
Single propagator	1450 (0.33%)	1895 (0.22%)	14 (0.95%)	24 (0.67%)
¼ sector propagators	5806 (1.34%)	9320 (0.87%)	56 (3.81%)	96 (2.67%)
¼ sector matchers	130392 (30.1%)	124828 (14.41%)	5 (0.34%)	288 (8%)
¼ sector total algo	**136198 (31.44%)**	**134148 (15.48%)**	**61 (4.15%)**	**384 (10.67%)**

¼ sector includes 4 propagator modules (each propagating 6 tracks per BX)
Matcher module handles 6 muons * 24 tracks = 144 objects
Matcher module resource utilization scales linearly with the number of objects = N_muons * N_tracks

Resource Usage on CTP7

Picture corresponds to table on previous slide (design for ¼ sector)

- **Blue: system firmware**
- **Red: track-muon matcher**
- **Yellow: track propagators**
- **Green: synchronizers & decoders**



Resource Usage for full sector

Extrapolated resource requirements for the full sector

Extrapolations were done starting from the resource usage for compiled firmware for ¼ sector on CTP7!

Percentages given w.r.t. Virtex Ultrascale+ VU9P FPGA

	LUT	FF	BRAM (36Kb)	DSP
Propagators	23224 (1.96%)	37280 (1.58%)	224 (10.62%)	384 (5.61%)
Matchers	2086272 (176%)	1997248 (84.5%)	80 (3.8%)	4608 (67%)
Total	2109496 (178%)	2034528 (86.1%)	304 (14.4%)	4992 (73%)

16 propagators, and matcher handles 24 muon * 96 tracks = 2304 objects

Important comments on extrapolation to Virtex Ultrascale+ VU9P FPGA

- **We will address the high resource usage (178%) in next set of studies**
- In this algorithm we are not doing any early cutting/sorting, so all muons are compared to all tracks of a given BX.
  - The resource usage grows quadratically, or proportional to N_Muons * N_Tracks.
  - It only takes up **30%** of CTP7 in quarter sector scenario, but grows by a factor of **16x** when we go up to a full sector, and doesn't fit even in Ultrascale+ FPGA
    - VU9P, which is a common choice for ATCA boards
- Another point is that we are using a **dR^2 cut** for matching in this algorithm, not a box cut
  - box cut would reduce the resources needed per Muon/Track combination

Possible future improvements in algorithm

There is room to both reduce the resource usage per Muon/Track combination, as well as limit the quadratic growth with these steps:

1) Add a **pre-matcher phase**, which uses a very crude comparison to select tracks that are in the same general area as the muon, and reject the rest early on. After that, the precise matching is done on every combination coming out of pre-matching stage. This will limit quadratic growth and allow for much better scaling.

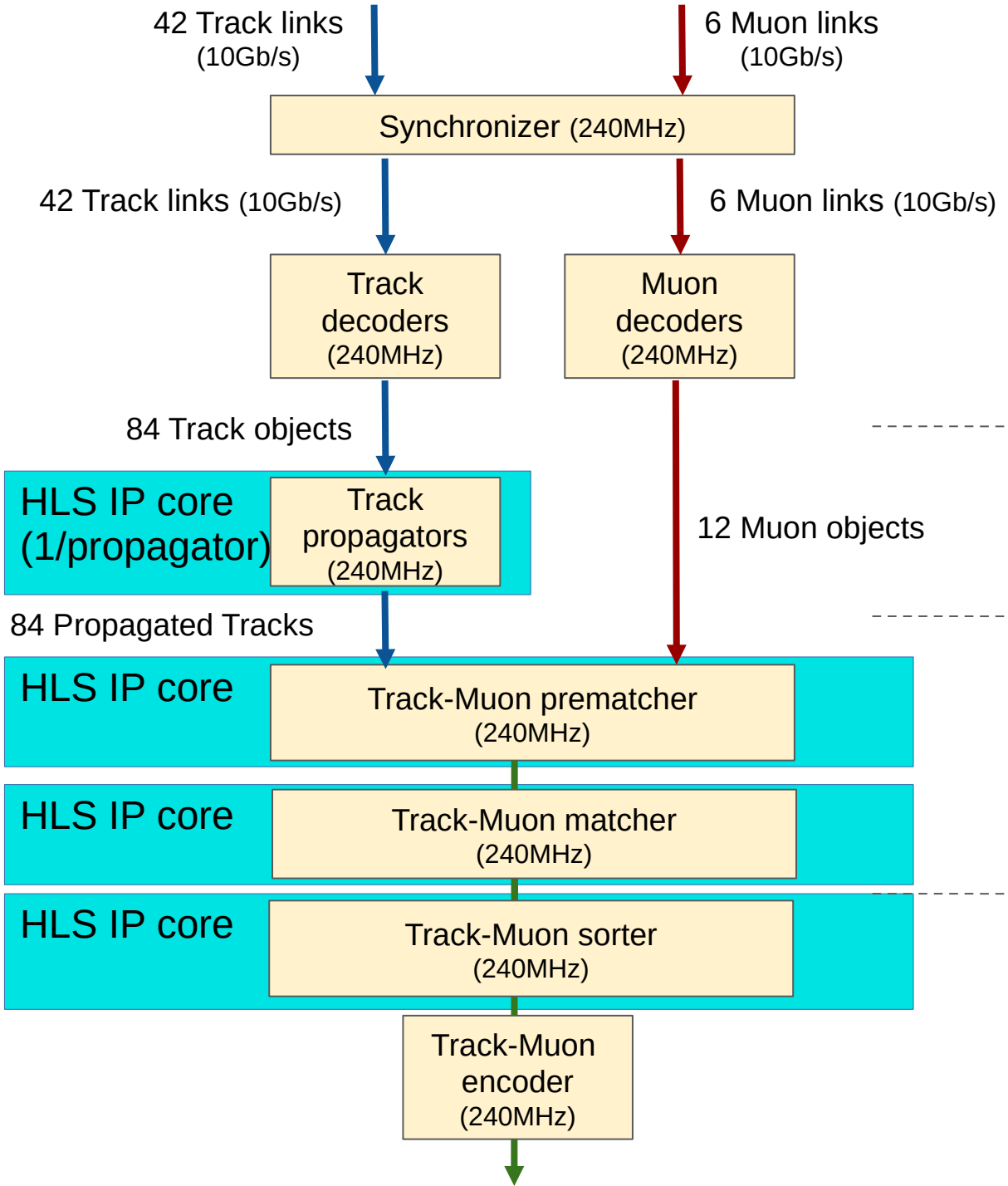
2) **Optimize the current matching method**: a box cut in phi/eta would probably do just as well as the dR cut and save resources per Muon/Track combination -- has to be checked

3) Going to **time-multiplexed design** would eliminate the need for including big overlap in the muon coverage (currently including +/- one muon sector for overlap)

4) If we don't go to time-multiplexed design, we can **drop some bits in phi**, because each board is limited to a sector view, which will help with resource usage

5) Possibly **increase the clock frequency** from 240MHz up to 320Mhz to allow more logic reuse per BX

Possible Track-Muon Correlator Design with pre-matching stage (With 48 fiber links)



Synchronization of input track and muon data
Input: 12 muons & 84 tracks (96 objects in total)

Decoding of Tracks and Muons
Output:
6 track objects per BX
6 muon objects per BX
(assumes 96-bit track/muon dataformat)

Track propagation to Muon station 2
Operates at 240MHz
One module propagates 6 tracks per BX
Need 14 modules for 84 tracks

Pre-matching: select best 3 matches in deta & dphi < 0.3
Matching: match track to muon if dR<0.2
Sorting: if more than one candidate in dR<0.2, pick candidate with best deltaPt
Output: 12 track-muon

Encoding of the track-muon objects

12 best Track-Muons

Short and long term plans

- Extensive testing of ¼ of sector firmware
  - Validation of firmware important next step
- Investigate possibility to use simplified transceiver logic
  - MP7 links use a lot of resources
    - Up to 30% for 48 fiber links on CTP7
- Enable a track-muon pre-matching step before the matching step
  - Potential to reduce the resource usage of the matcher by limiting number of dR^2 calculations
- Use MTF7 board as the source for muons for the CTP7
  - This would allow to scale to the full sector
    - 48 links for tracks (still in loopback with CTP7) and 12 links for muons from MTF7
    - Total of 120 objects per BX: 96 tracks + 24 muons
- Need Florida group help for further development
  - E.g. to optimize dR matching window
- Implement track-veto module for the displaced muon
- Synthesize full scale sector targeted for ultrascale device

Summary

- Hardware implementation of prototype correlator shown to work
  - With realistic conditions: continuous data received over fiber links & processed with deterministic latency
- Resource usage is reasonable
  - ¼ of trigger sector fits on CTP7 FPGA
- Currently working on extrapolating estimates to full scale of correlator sector
- Lot of ideas can be explored to improve the track-muon correlator design