Workshop on a Future Muon Program At Fermilab

Mu2e-II TDAQ developments

G. Pezzullo, A. Gioiosa
Yale University, Univ. del Molise/INFN

Introduction

- Requirements for the Mu2e-II DAQ?
- Mu2e-II will have more beam on target and higher granularity detectors.
- Assumptions:
  - Power and cooling limitations are solved by money
  - Installation around 2030
  - Control and Synchronization of the detector will work itself out, this talk focuses on Trigger and Data Paths
- This talk outlines the ideas that were proposed and

Implications

- ~2x more detector channels, and ~5x more pulses on target, for ~10x higher data rate (if background remains the same)
  - Current expected Mu2e-I data rate from front-ends is 40 GBps
- More detector channels and more background implies bigger event sizes (maybe ~3x?)
  - Mu2e-I expected event size is 200KB
- Tape capacity for Mu2e-I is 7PB/year
  - Might assume 2x increase for Mu2e-II to 14PB/year
- Necessary rejection for Mu2e-II is ~3000:1
  - 600 KB events @ 3 MHz -> 560 MB

Implications

- Reduced OFF Spill periods (to no OFF Spill time?) implies less advantage for large front-end buffers streaming data:
  - In Mu2e-I, have second of downtime to play catchup
  - In Mu2e-II, steady event rate (could buffer just to handle event to event variation, not large accelerator time structures)
- No large front-end buffers at CRV would imply need for low-latency trigger decision for CRV.
  - Low latency trigger decision implies an FPGA trigger layer.
- Consider the cost of these scenarios:
  - Large CRV buffers and software trigger
  - Small CRV buffers and hardware trigger

Streaming vs Triggered

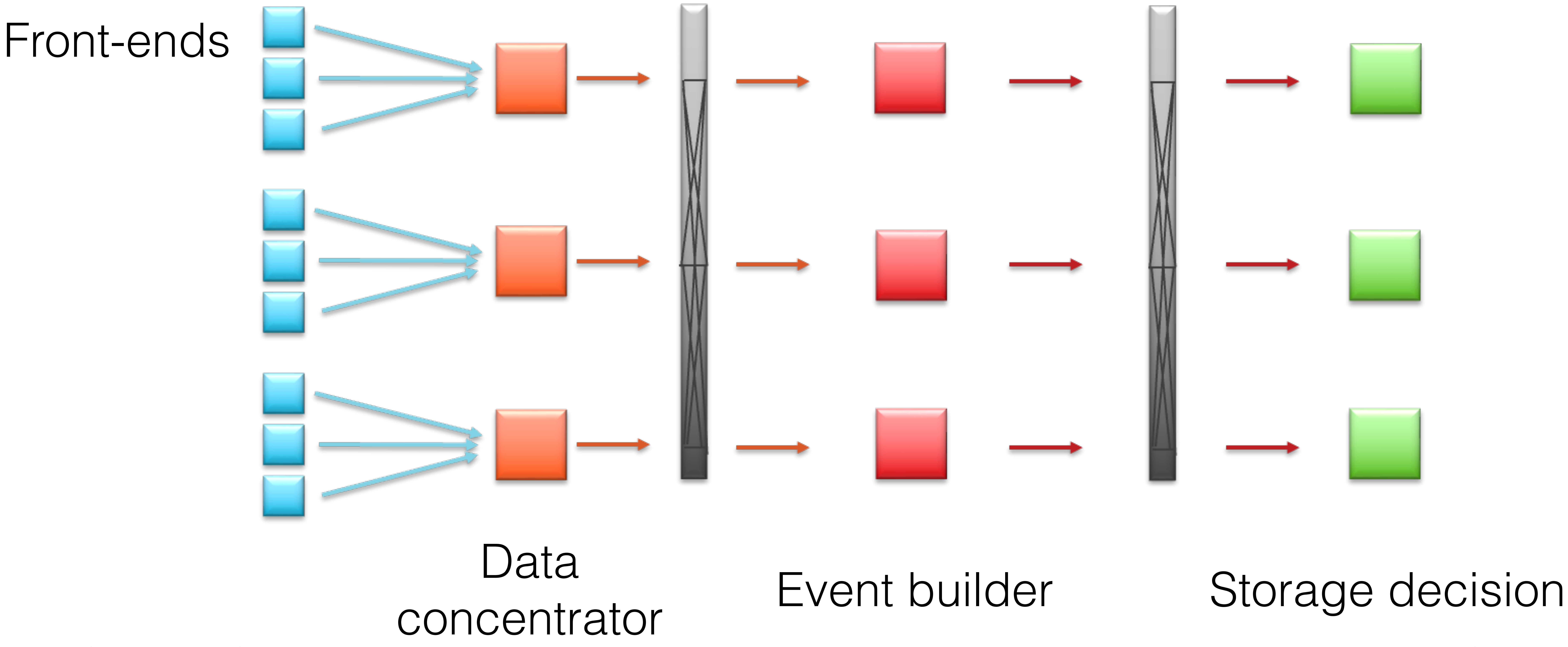
- Important upfront decision as to which detector subsystems are triggered.
- Same as Mu2e-I?
  - Stream all Tracker and Calorimeter data
  - Software Trigger for CRV based on Tracker and Calorimeter
- Alternatives:
  - Stream Calorimeter Data
  - Hardware Trigger for Tracker and CRV based on Calorimeter
  - High-level Software Trigger for storage decision

Radiation Tolerance requirements

- Radiation levels at the detector will be higher than Mu2e-I
  - Mu2e-II comparable to Calorimeter level of CMS phase-II?
- For Mu2e-I, using the VTRx was a primary constraint
  - We had to change the DAQ topology as a result
- Mu2e-II likely will not want to design their own rad-hard links, so we will be at the mercy of CMS/ATLAS
  - This should be worked out as soon as possible

Generic Data Readout Topology

- Multi-stage TDAQ system

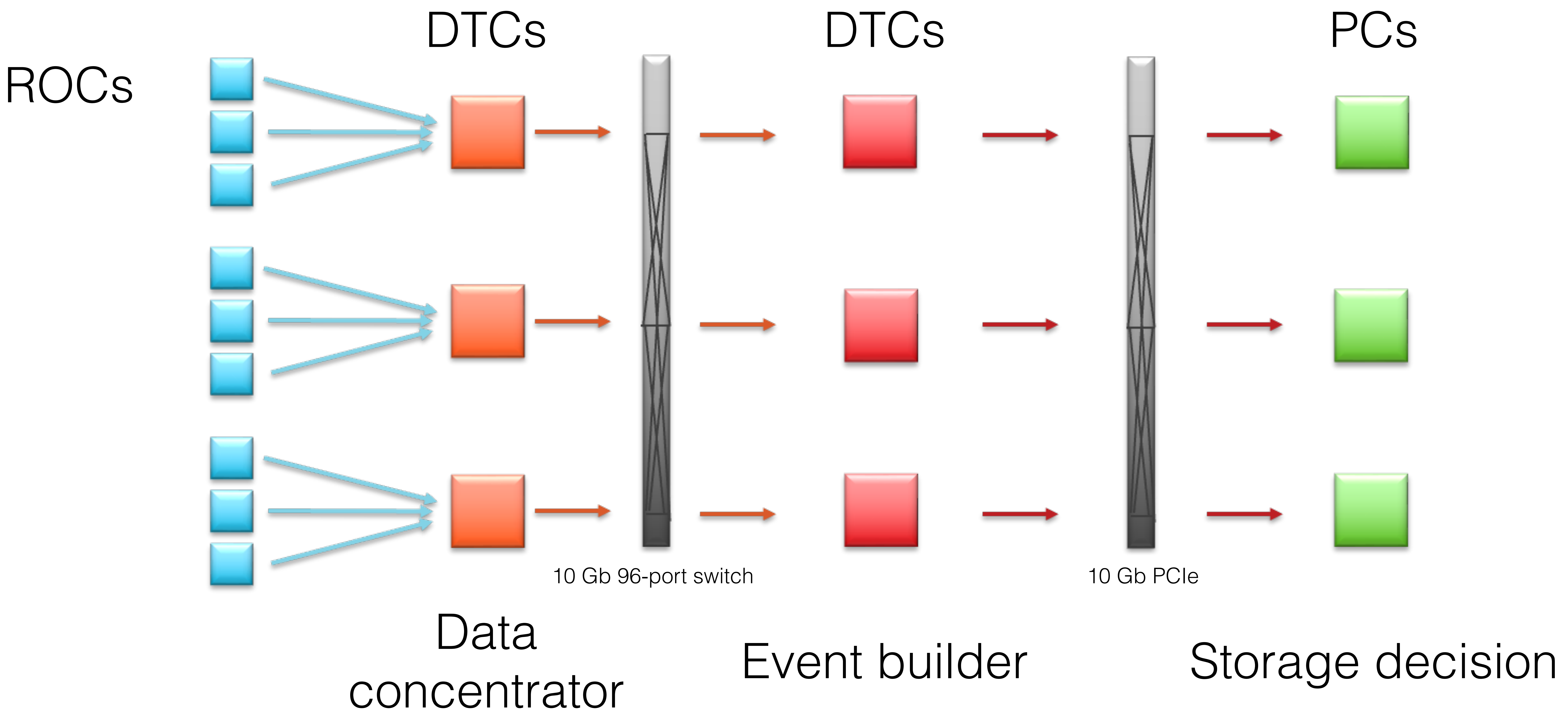


Generic Data Readout Topology

- **Data Concentrator:**
  - Aggregate small front-end fragments into larger chunks for efficient event building
- **Event Builder:**
  - Data is switched from Concentrator Layer to Event Builder Layer such that full events arrive at Event Builder Layer and are buffered.
    - Preprocessing or filtering could occur
- **Storage Decision:**
  - Available decision nodes make high level storage decision on full events retrieved from Event Builder Layer buffer

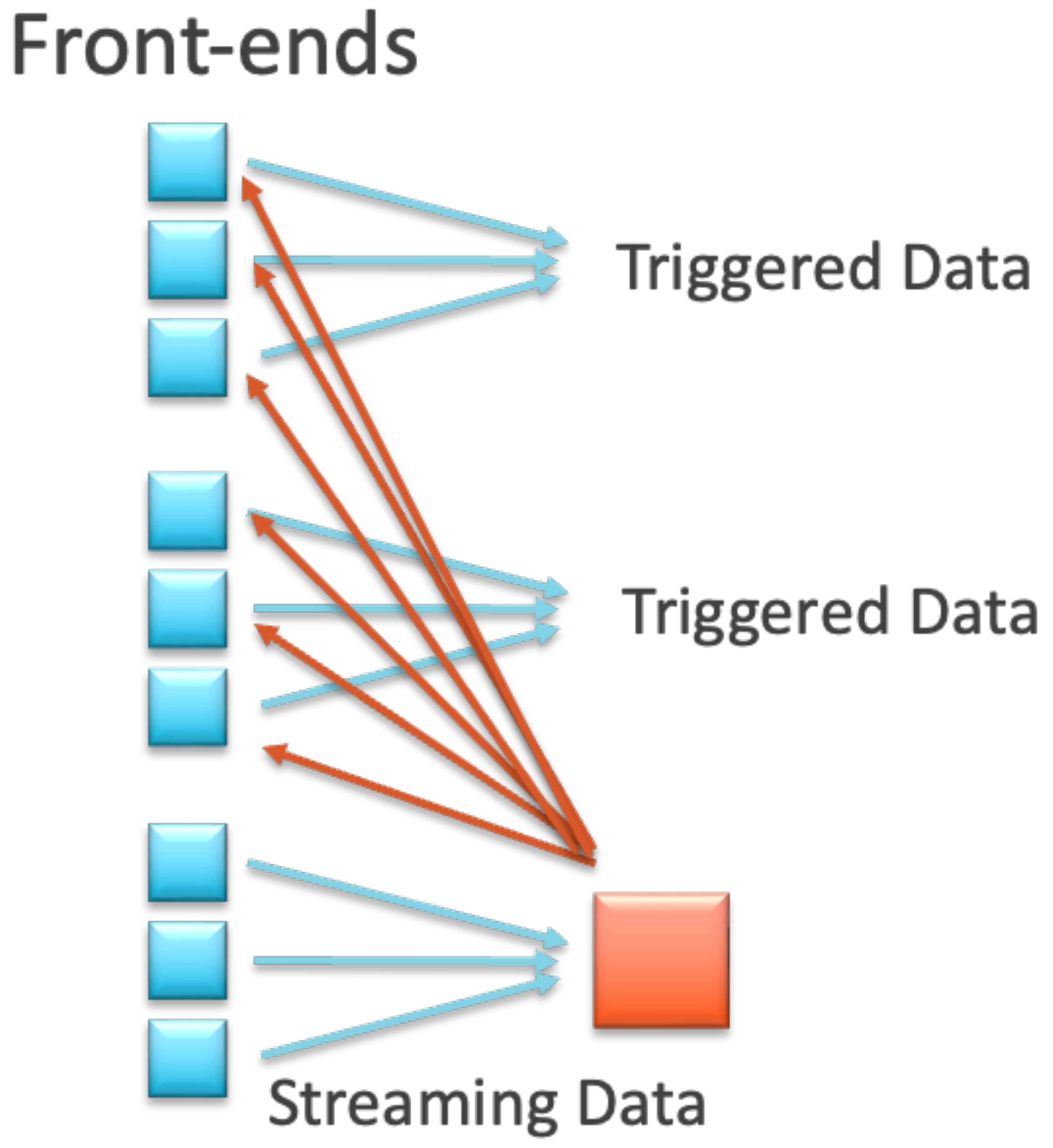
Generic Data Readout Topology applied to Mu2e-I

- The Mu2e case



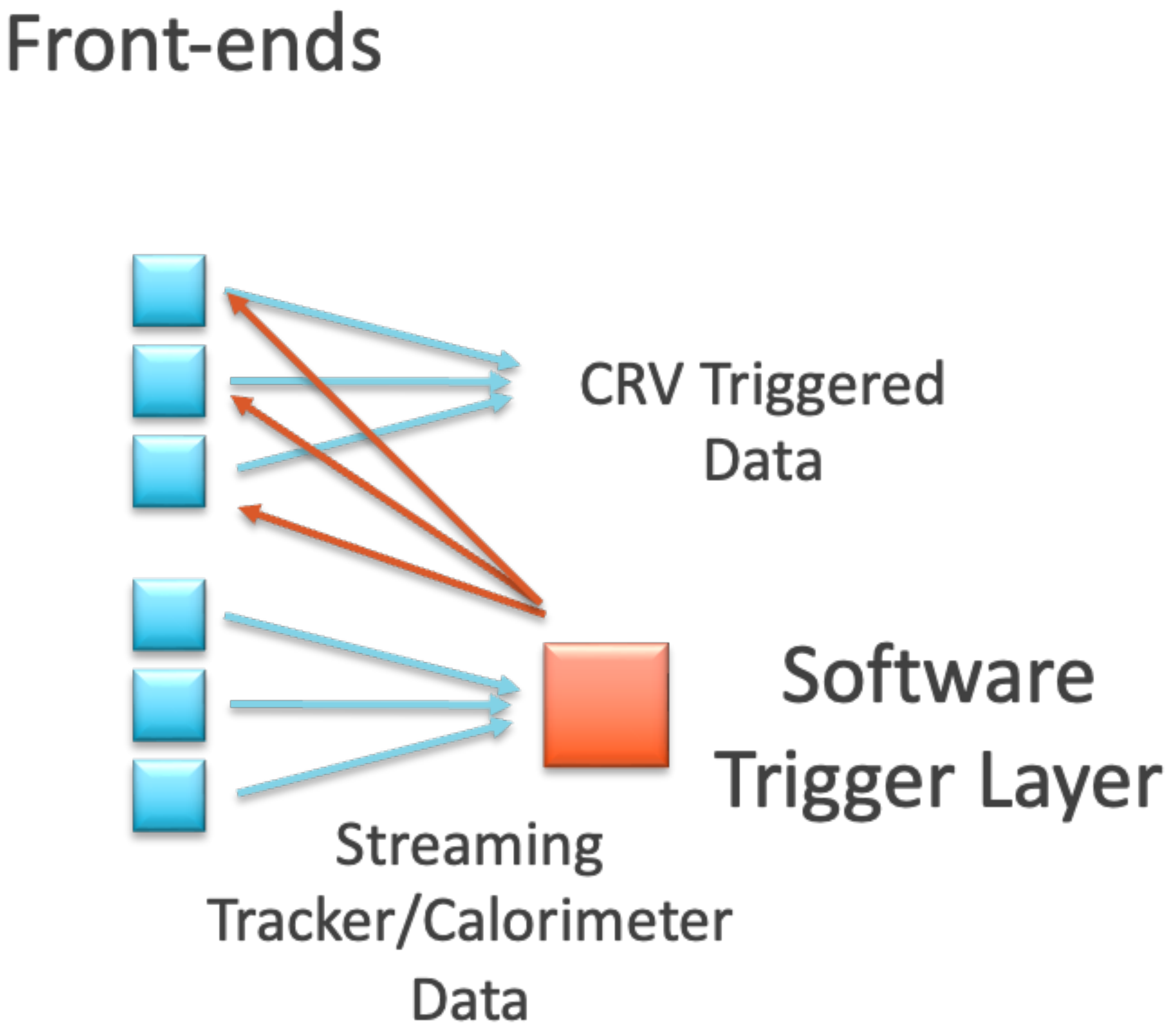
Generic Data Readout Topology

- Data transfer can be minimized by:
  - transferring only trigger-primitives
  - pulling all the data only for triggered events



Generic Data Readout Topology applied to Mu2e-I

- In Mu2e, we use this approach already in the second stage of the event-filtering (after the trigger decision is made already) for pulling the CRV data



TDAQ LOIs from Snowmass 2021

- **A 2-level TDAQ system based on FPGA pre-processing and trigger primitives**
  - ROCs (create trigger primitives, buffer event fragments), L1 FPGA layer (getting trigger primitives from calo and tracker), and HLT layer (requests event fragments from full detector)
- **A 2-level TDAQ system based on FPGA pre-filtering**
  - Leverage HLS for FPGA rejection
- **TDAQ based on GPU co-processor**
  - Using GPUs at HLT (or L0)
- **A trigger-less TDAQ system based on software trigger**
  - Scale up current system

TDAQ LOIs from Snowmass 2021

- Serious implications in the TDAQ-farm room requirements (not enough cooling if we would use the current Mu2e TDAQ room)
- Data transfer and processing become very challenging

- **A trigger-less TDAQ system based on software trigger**
  - Scale up current system

TDAQ LOIs from Snowmass 2021

- Data transfer is not trivial
- Importing C-style algorithm is not simple

- **TDAQ based on GPU co-processor**
  - Using GPUs at HLT (or L0)

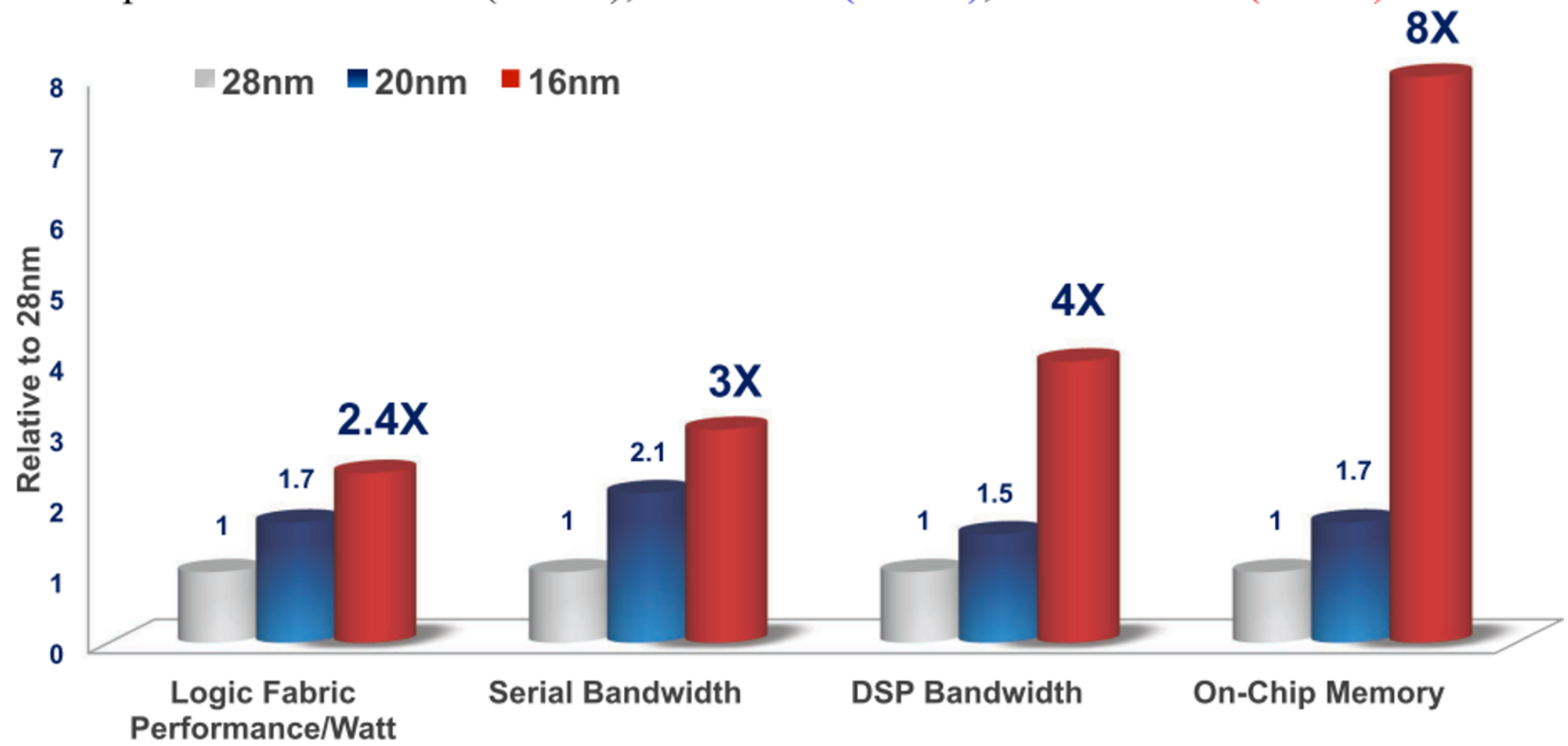
TDAQ LOIs from Snowmass 2021

- **A 2-level TDAQ system based on FPGA pre-processing and trigger primitives**
  - ROCs (create trigger primitives, buffer event fragments), L1 FPGA layer (getting trigger primitives from calo and tracker), and HLT layer (requests event fragments from full detector)
- **A 2-level TDAQ system based on FPGA pre-filtering**
  - Leverage HLS for FPGA rejection

- FPGA can offer flexibility for algorithm development
- Mu2e is already using FPGAs in the ROCs and the DTCs
- These solutions are more tight to the sub-detector readout systems

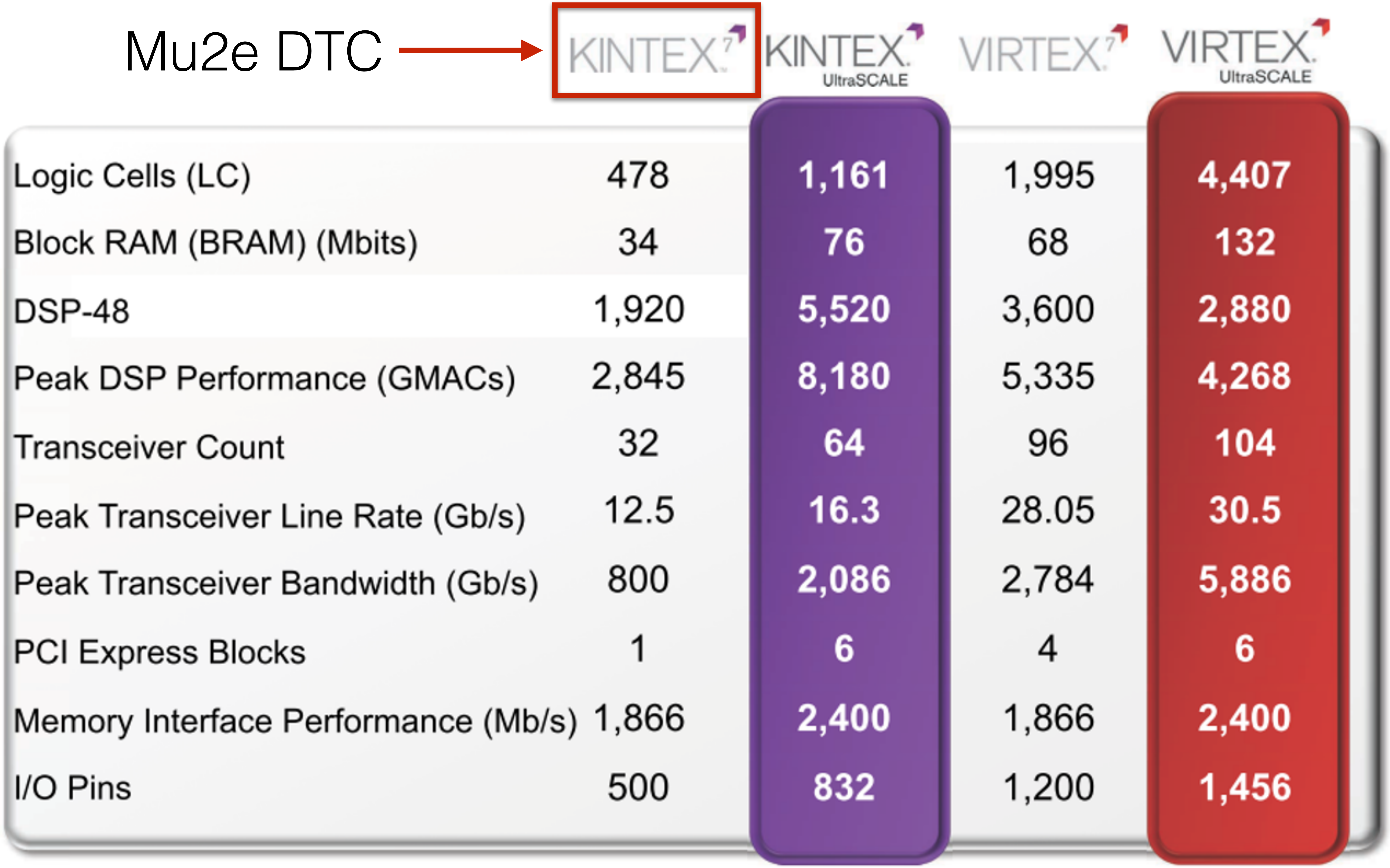
FPGA scaling

Example: Xilinx Virtex 7 (28 nm), Ultrascale (20 nm), Ultrascale + (16 nm)



FPGA scaling

Mu2e DTC →

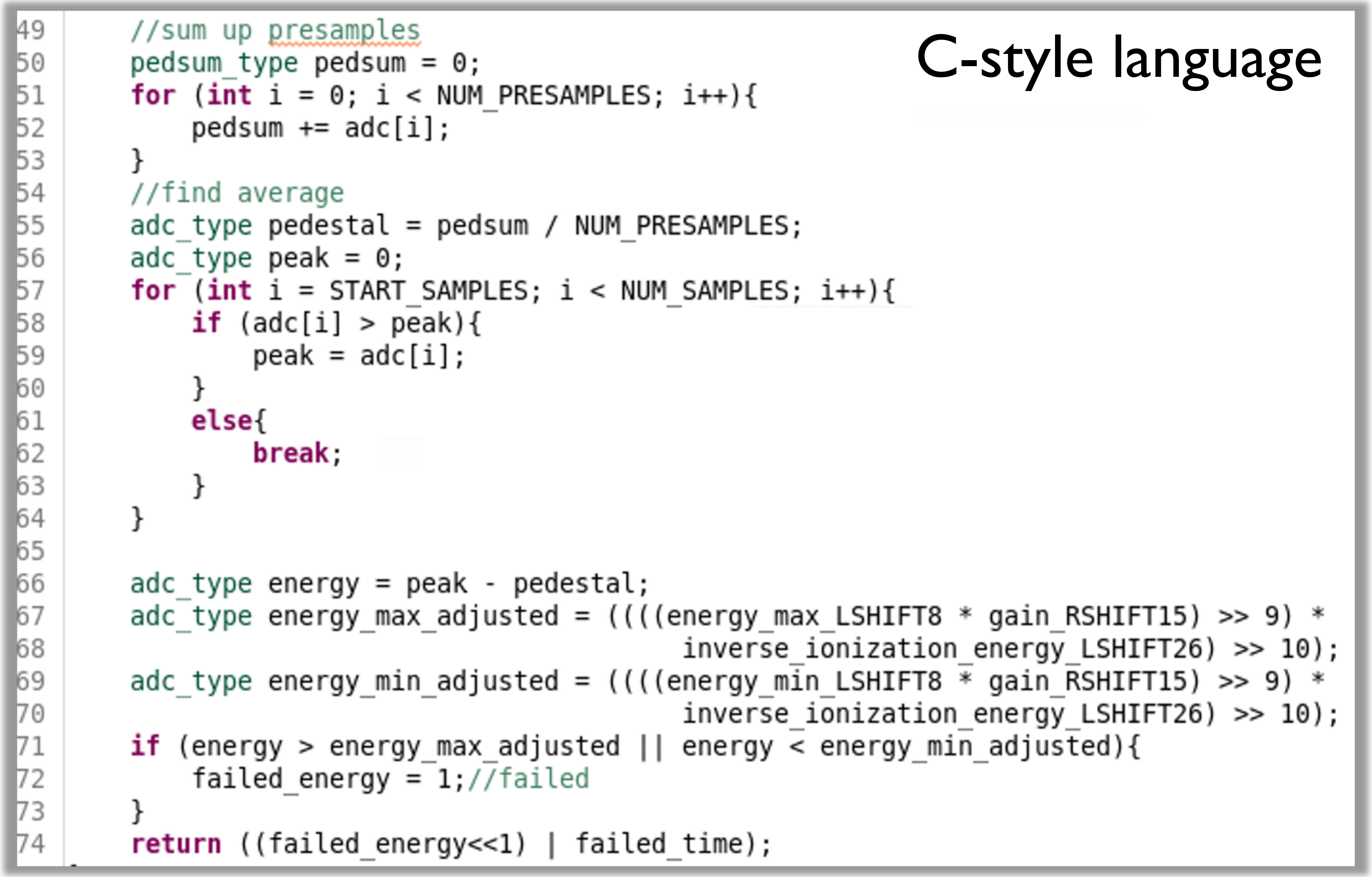


	KINTEX 7	KINTEX UltraSCALE	VIRTEX 7	VIRTEX UltraSCALE
Logic Cells (LC)	478	1,161	1,995	4,407
Block RAM (BRAM) (Mbits)	34	76	68	132
DSP-48	1,920	5,520	3,600	2,880
Peak DSP Performance (GMACs)	2,845	8,180	5,335	4,268
Transceiver Count	32	64	96	104
Peak Transceiver Line Rate (Gb/s)	12.5	16.3	28.05	30.5
Peak Transceiver Bandwidth (Gb/s)	800	2,086	2,784	5,886
PCI Express Blocks	1	6	4	6
Memory Interface Performance (Mb/s)	1,866	2,400	1,866	2,400
I/O Pins	500	832	1,200	1,456

FPGA algorithm development: HLS

- High Level Synthesis is now good enough to rival manual VHDL or Verilog algorithm development
- Allows physicists to easily understand and develop low and fixed latency FPGA algorithms
  - Makes emulation easy for offline
- Debug and verify in a software environment (often 10x faster iterations than firmware simulation tools)
- CMS is heavily investing in HLS approach to FPGA algorithm development.
  - There is a hls4ml collaboration developing machine learning (neural network) tools using HLS

Coding in HLS



C-style language

```
    //sum up presamples
    pedsum_type pedsum = 0;
    for (int i = 0; i < NUM_PRESAMPLES; i++){
        pedsum += adc[i];
    }
    //find average
    adc_type pedestal = pedsum / NUM_PRESAMPLES;
    adc_type peak = 0;
    for (int i = START_SAMPLES; i < NUM_SAMPLES; i++){
        if (adc[i] > peak){
            peak = adc[i];
        }
        else{
            break;
        }
    }

    adc_type energy = peak - pedestal;
    adc_type energy_max_adjusted = ((((energy_max_LSHIFT8 * gain_RSHIFT15) >> 9) *
                                      inverse_ionization_energy_LSHIFT26) >> 10);
    adc_type energy_min_adjusted = ((((energy_min_LSHIFT8 * gain_RSHIFT15) >> 9) *
                                      inverse_ionization_energy_LSHIFT26) >> 10);
    if (energy > energy_max_adjusted || energy < energy_min_adjusted){
        failed_energy = 1;//failed
    }
    return ((failed_energy<<1) | failed_time);
```

Why multi-staged TDAQ?

- From Mu2e studies, we know that >70% of the hits produced in the tracking detector is made by very low-P (<10 MeV/c) e
  - Identifying them is possible
  - If we can identify these hits, we can suppress them and reduce the data throughput by quite a lot
    - ML tools are available on FPGA!
- In principle, the Helix patter-recognition can be coded on FPGA
- One could use **very** powerful FPGAs if we locate them outside of the detector solenoid

Proposed R&D strategy

- The majority of the people involved with the group is quite busy developing the Mu2e TDAQ system
    - We need to create additional "expertise" on algorithm development on FPGA
- Use the current Mu2e trigger algorithms to perform feasibility studies
    - Development can happen with commercial boards
- A successful demonstration will consist of delivering a demonstrator that can be plugged-in parasitically in the Mu2e TDAQ towards the end of the Run-2