FELIX readout status

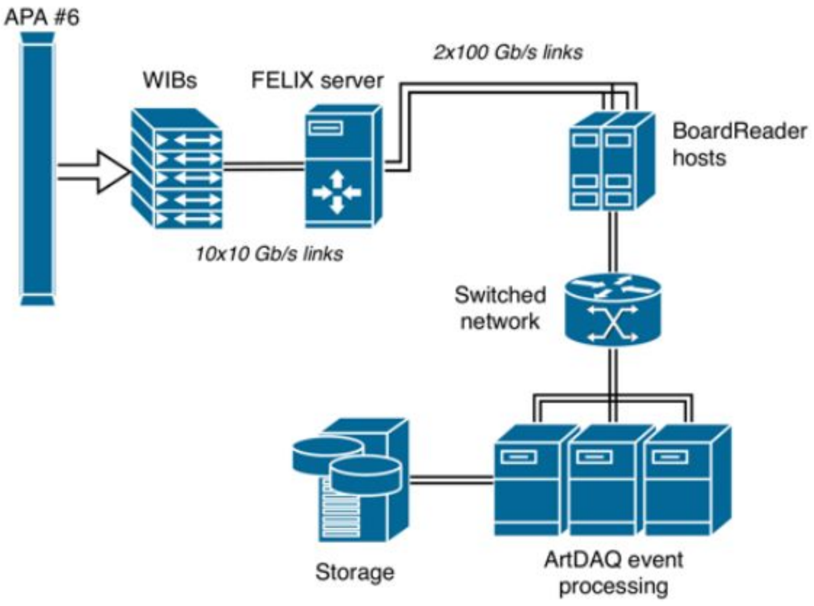
DAQ Workshop
2019-02-04

Roland Sipos
CERN EP-DT

FELIX recap

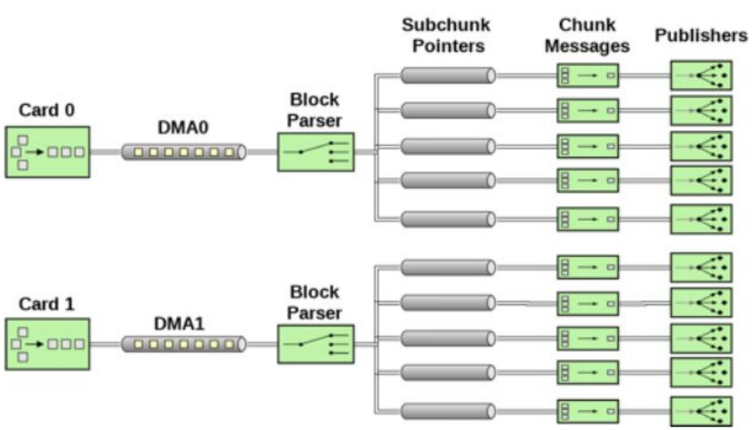
The Front-End Link eXchange is the upgraded readout system of the ATLAS experiment :

- Approach relying on servers and COTS to do data processing
  - PCIe based FPGA custom card
  - Networked scalable system in ProtoDUNE Single Phase
- FELIX firmware and software tuned for specific use case
- Software trigger matching
- Software compression
  - Hardware accelerated with Intel® QuickAssist Technology (QAT)



Modifications for ProtoDUNE

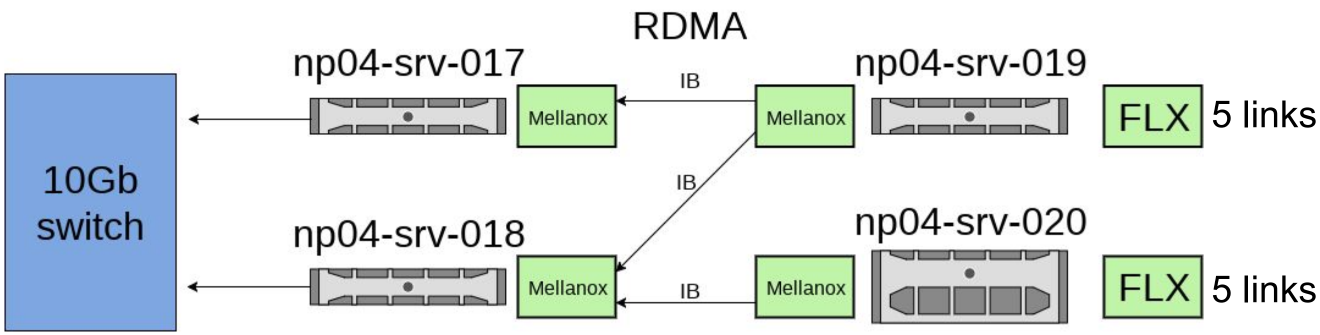
- In order to sustain the high rate of incoming frames (2 MHz) and high throughput requirements, modest modifications of the firmware were introduced:
  - Chunks are packed together (x12) in order to minimise memory-copy effort at the publisher software level. Rate of networking calls is also greatly reduced.
    - 2MHz of 464B -> 166kHz of 5568B
  - DMA payload (block) size increased (1 ->4 kB) in order to optimise parsing.
- FELIX publisher - felixcore:
  - Removal of block and chunk copy pipelines



- **In ProtoDUNE only uni-directional traffic from detector** is used and data fragments have a fixed size
- **Optimised** in order to achieve the required data throughput:
  - Simplified data routing by means of a **dedicated threads per physical link**
  - ***Infiniband libfabric used for networking***

Topology for beam run

APA6 readout



- We started off with default DAQ servers (Dell R730 , dual 8 cores @2.1 GHz)
  - Infamous “10 links problem”: Discovered that the present servers are not capable (memory bus limitation) to support the full load of one APA. 7 out of 10 links can be run stably.
  - Resorted to using 2 FELIX servers, which ran very stably and successfully throughout beam data taking
- FELIX is a high I/O performance application
  - Good knowledge of tuning techniques is important
  - Choosing optimized software and I/O protocols is vital
- With the hardware we had at hand we decided to not test any merging of FELIX and BR functionality

New FELIX hosts

2 new servers for hosting FELIX cards (srv-025, srv-026)

Intel(R) Xeon(R) Gold 6136 CPU @ 3.00GHz

- # of Cores 12
- # of Threads 24
- Processor Base Frequency 3.00 GHz
- Max Turbo Frequency 3.70 GHz
- Cache 24.75 MB L3
- # of UPI Links 3
- TDP 150 W

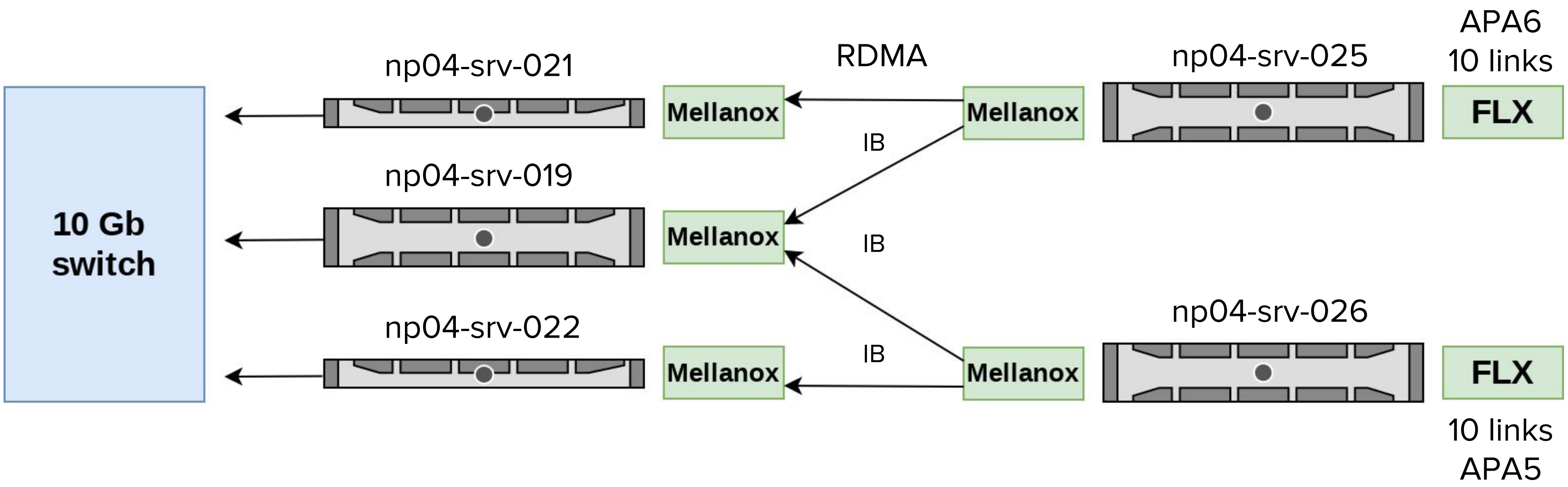
BSIT-2UHPS-DS

Dual socket P (LGA 3647) supports Intel® Xeon® Scalable Processors, Dual UPI up to 10.4GT/s
Up to 2TB ECC 3DS LRDIMM, up to DDR4-2666MHz; 16 DIMM slots
6 PCI-E 3.0 x16 slots, 1 PCI-E 3.0 x8 (in x16) slots
8 Hot-swap 2.5" SAS/SATA drive bays, 2 Hot-swap 2.5" SAS/SATA/NVMe drive bays
1 SIOM for flexible Networking
5x 8cm heavy duty fans with optimal fan speed control
2000W Redundant Power Supplies Platinum Level (94%)

And it worked like a charm

- 10 links sustained over several days without need for restarting felixcore
- Discussion with Intel experts
  - Main bottleneck understood: DELL machines memory throughput problems
    - STREAM memory benchmark
- Bottom line: FELIX is sensitive to hosting servers' specifications. Keep and eye on:
  - benchmark of CPUs,
  - maximum memory throughput,
  - NUMA setup of MOBO.
    - Interrupt affinities
    - Thread core pinning

New topology



- APA5 moved to FELIX readout
- Mellanox boxes are ConnectX-5 100Gb NICs.
- 019 is connected to the chain for testing and performance optimization

BR optimization

- New netio version ensures the STOP-START capability
  - Already integrated and it works. On extremely rare occasions, some instability with re-subbing
- Sustaining >40Hz of trigger rate is problematic
  - Either memory throughput limit or QAT load balancing of compression.
  - Statistics shows increased compression time required using the embedded QAT solution
  - Under investigation (also communicating with Intel experts)
  - Some driver/software tweaks already improved the situation a bit
  - NIC and QAT sits on same NUMA node (but we need a riser to move the NIC)
- Cleaning up code-base
  - BR relies on 3 main dependencies (Netio high level network lib, libfabric, QAT driver)
    - Fabric and QAT driver should be a UPS product (not so simple for QAT)
  - Preparing documentation
  - Automation aspects -> Ansible role for full publisher and BR host setup

Main objective - short term

Utilize a single host's capabilities as much as possible

- Trigger primitive finding in software, using AVX registers
  - Phil has a functional application to do hit finding
    - Integration to the BR code is done
    - Performance optimization is ongoing
  - Self triggering in ProtoDUNE is not so far from reality
- Running data processing on FELIX host
  - A BR version that directly reads and processes data from the FELIX DMA circular buffers
  - Initial tests shows high cost for moving around data in memory (queueing in)
    - 1 CPU (12 cores) is almost fully used (~85% CPU utiliziation)
    - Few more "tricks" are under investigation
      - Don't "serialize-then-move" data, but scatter-gather chunks into destination buffer
      - Exploit NUMA locality: FELIX node and associated cores are only responsible for chunk parsing and stream continuous WIB frames to the other NUMA node for data processing.
    - Some additional FW changes could result with substantial gain for reducing CPU/memory bandwidth requirements

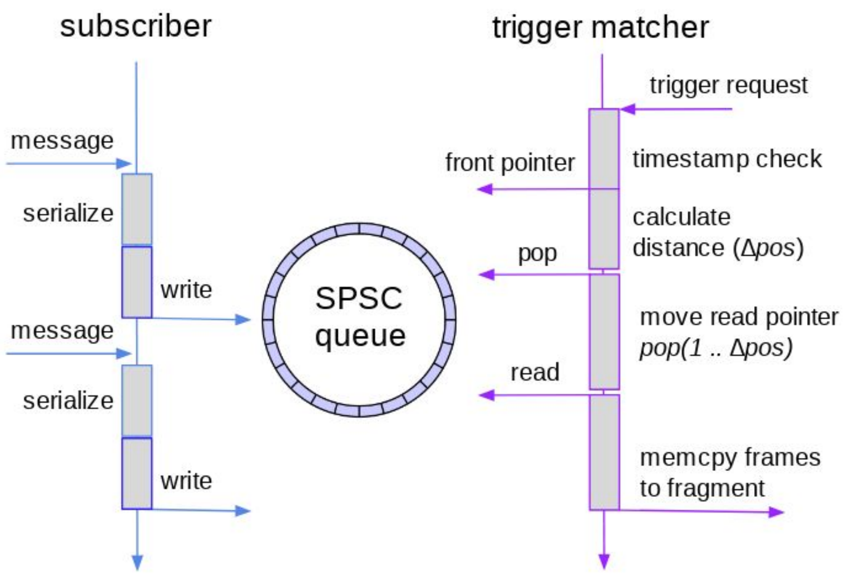
Summary

- ProtoDUNE DAQ has reached a remarkable quality considering the project timescale and the “recycling approach” taken to meet deadlines
  - It is not comparable to DUNE in many aspects
  - It allows a playground to develop ideas for DUNE (2019++)
- We got away with loose ends and shortcuts that would be show stoppers, if they were repeated in DUNE
  - A much more rigorous specifications/design work has to be carried out
  - ProtoDUNE allowed to build up a great team: if we keep the momentum (and know how) all elements are there to make the DUNE DAQ

Backup slides

BoardReader implementation

- The FELIX BoardReader is implemented as part of the artdaq data acquisition framework
- Receives and buffers data continuously
  - One subscriber thread for each link populating a SPSC queue (lock-free implementation from the Folly library)
  - A specialized thread extracts data from buffer, matching a 5 ms time-window based on the trigger request from Event Builders at ~25Hz (baseline rate, achieved 60 Hz)
  - Re-ordering and compression of data
  - Complete fragment with compressed data is sent downstream to EventBuilders



Fragment compression

- ProtoDUNE target compression factor set to 4 (implications in storage hardware projections)
  - Efficient compression can be achieved by re-ordering the frames to contiguous ADC data for individual channels
- Compression **should also keep up with the 25 Hz trigger rate and the about 46 Mbytes payload size**
- **Hardware accelerated compression**
  - Intel® QuickAssist Technology (QAT)
  - Under study at CERN
  - Can offload the CPU and compress faster
  - DEFLATE algorithm
  - Intel® Xeon® Scalable processors with integrated QAT support used in BoardReader hosts
    - C628 chipset
  - **Allows a reduction of the time required for the compression of one data fragment**
    - from about 100-200 ms (software only)
    - to about 5-9 ms (accelerated)