Entry Stage for the CBM First-level Event Selector

Dirk Hutter
for the CBM Collaboration

hutter@compeng.uni-frankfurt.de
FIAS Frankfurt Institute for Advanced Studies
Goethe-Universität Frankfurt am Main, Germany
The CBM Experiment at FAIR

- Fixed target heavy ion experiment at FAIR
- Physics goal: exploration of the QCD phase diagram
- Extreme reaction rates of up to 10 MHz and track densities up to 1000 tracks in aperture
- Conventional trigger architecture not feasible
- Full online event reconstruction needed

➡ Self-triggering free-streaming readout electronics
➡ Event selection exclusively done in FLES HPC cluster
First-level Event Selector (FLES)

- FLES is designed as an HPC cluster
  - Commodity PC hardware
  - First phase: full input connectivity, but limited processing and networking

- FPGA-based custom PCIe input interface
  - Total input data rate > 1 TB/s

- Timeslice building via InfiniBand network
  - Combines the data from all input links to self-contained overlapping processing intervals and distributes them to compute nodes
  - RDMA data transfer very convenient for timeslice building

- Located in the Green IT Cube data center
  - Cost-efficient infrastructure sharing
  - Maximum CBM online computing power only needed in a fraction of time
    → combine and share computing resources

Consequences
- Transmit 10 TBit/s over 700 m distance
- Needs single-mode optics
- Increased link latency
Initial CBM DAQ/FLES Architecture

- **Single flat cluster** design
- Two FPGA-based stages
  - Data Processing Board (DPB): aggregation and preprocessing
  - FLES Interface Board (FLIB): data preparation and interface to standard COTS hardware
- Long-range connection to Green Cube via **custom optical links**
- Revisited design in context of CBM modularized start version and usage of GBTx frontends
  - Consider future **upgradability**
  - Maximize use of **standard hardware**
Optimized DAQ/FLES Architecture

- **Combine DPB and FLIB** to single FPGA board, similar to ATLAS, LHCb and ALICE
- Split computing into 2 dedicated clusters
  - Only small entry cluster holds custom hardware
  - Relaxed requirements on processing cluster design
- Long-range connection to Green Cube via **standard network** equipment (e.g., long-range InfiniBand)
FLES Entry Stage Design

- **Input from detector systems**
  - 4800 GBT links @4.48 GBit/s (21.5 TBit/s total bandwidth)
  - < 50% link occupancy → 10 Tbit/s peak
  - **100 CRI** with 48 links and PCIe 4.0 x8 (3.0 x16) “100 GBit/s card”

- **Output to processing cluster**
  - No significant data reduction in entry cluster possible
  - Averaging in spill intensity variations in entry stage can smooth data rate
  - Averaging machine duty cycle foreseen in processing cluster
  - Connectivity to ~ 600 processing nodes
  - Absolut worst case → 10 Tbit/s peak

- **Possible entry node configuration**
  - 2 CRI + 2 HCAs per node feasible
  - 50 nodes @200 Gbit/s or 100 node @100 Gbit/s

CRI prototype candidates:
- FLX-712
- HTG-Z920

Current entry node prototype:
ASUS ESC8000
Reminder: InfiniBand Networks

- High-throughput, low latency switched fabric network
  - Widely used in HPC applications
- RDMA semantics perfectly suitable for FLES timeslice building
- Lossless fabric, i.e. packets are not routinely dropped
  - Credit based flow control on link-level
  - Each (logical) link supplies credit to the sending device denoting the available buffer space
- Quality of Service features
  - Virtual lanes (VL) are separate logical communication links that share a single physical link
  - Up to 15 virtual lanes (VL) and one management lane per link
- Next generation: HDR (200 Gbit/s)
  - 40 port switches, e.g. QM87xx series
  - Allows port splitting → HDR100, 80 ports per switch
Long-haul connections

• Challenge: bridge 700m between entry and processing cluster

• Physical communication
  • Typical intra data center techniques reach 100-300 m
  • Use medium or long range single-mode optics, e.g. CWDM4 up to 2km

• Protocol
  • All reliable communication protocols suffer from high bandwidth-delay products
  • Higher delays account for more buffers or lower bandwidth

• Mellanox InfiniBand switches allow to switch off VLs
  • Collapses buffers form unused VLs into larger buffer
  • Idea endorsed by Mellanox

• To be tested in real lab setup
Long-haul Connection Test Stand

- Testing requires connection with given delay
- Box with 8x 400m + 8x 800m single mode fiber
  - G.657.A1 bend-insensitive fiber to allow compact size
- Each box is sufficient for:
  - 1 PSM4 connection
  - 4 LR4 / CWDM4 connections using MPO-LC breakouts
- Allows measurement in 400m steps
  - e.g. 400m, 800m, 1200m until patching exceeds loss budget, ~3,5 dB
- First lab tests with InfiniBand EDR very promising
Possible FLES Network

- Fat-tree like network
  - Sub-clusters share a common spine
  - Compute cluster can have 1/6 blocking ratio

- Timeslices are built across the long range connection

- Local spine for minimal connectivity w/o compute cluster
Network with CBM-local Timeslice Building

- Build timeslices to EN memory → send fully built timeslices to GC
  - Additionally needed network resources feasible with HDR

- Timeslice building decoupled from GC network
  - More control over critical building process, no long link delay
  - Independent of GC network architecture, flexible against changes
  - Better scaling to more PNs

- Infiniband routers additionally allow to separate subnet control

- 20 Tbit/s local (shared)
- 12 Tbis/s inter cluster
- Up to **100 ENs** @ 100Gb/s in
- 9 + 9+ switches, 100 + 60 cables

- 20x HDR
- 100 EN nodes
- HDR
- 120 fibers
- < 600m OS2
- 120 Tbit/s local (shared)
- 12 Tbis/s inter cluster
- Up to **100 ENs** @ 100Gb/s in
- 9 + 9+ switches, 100 + 60 cables

- 20 Tbit/s local (shared)
- 12 Tbis/s inter cluster
- Up to **100 ENs** @ 100Gb/s in
- 9 + 9+ switches, 100 + 60 cables

- 120 fibers
- < 600m OS2
- 120 Tbit/s local (shared)
- 12 Tbis/s inter cluster
- Up to **100 ENs** @ 100Gb/s in
- 9 + 9+ switches, 100 + 60 cables
Thanks for your attention
Local TS Building + Application Level Routing

- Use separate HCA (port) instead of TS building network to send timeslices to GC
- Full separation between networks
  - Mix of different network technologies possible
- Less flexible in terms of link balancing to GC

120 EN nodes
2x HDR100

- Application level routing
- 12 Tbit/s local
- 12 Tbis/s inter cluster
- Up to 120 ENs @ 100 Gb/s in
- 8 + 9+ switches, 60 + 60 cables

120 fibers
< 600m OS2
FLES Data Management Framework

- RDMA-based timeslice building (*flesnet*)
  - Works in close conjunction with FLIB hardware design

- Paradigms:
  - Do not copy data in memory
  - Maximize throughput

- Based on microslices, configurable overlap

- Delivers fully built timeslice to reconstruction code

- Prototype implementation available
  - C++, Boost, IB verbs
  - Measured flesnet timeslice building (8+8 nodes, including ring buffer synchronization, overlapping timeslices):
    - ~5 GByte/s throughput per node

- Prototype software successfully used in several CBM beam tests

Diagram:

- FLIB Server
- Device Driver
- SHM
- TS-Building IN
- IB Verbs
- TS-Building CN
- IB Verbs
- HCA
- HCA
- SHM
- Reco/Ana

- Direct DMA to InfiniBand send buffers
- Shared memory interface
- 10 GBit/s custom optical link
- Timeslice building
- InfiniBand RDMA, true zero-copy
- Indexed access to timeslice data

Legend:
- FEE
- FLIB
- SHM
- IB Verbs
- HCA
**RDMA Timeslice Building**

- **Two pairs of ring buffers for each input link**
  - Second buffer: index table to variable-sized data in first buffer

- **Copy contiguous block of microslices via RDMA (exception: borders)**

- **Lazy update of buffer status between nodes, reduce transaction rate**
FLES Input Data Path

- Full offload DMA engine
- Transmit microslices via PCIe/DMA directly to userspace buffers
  - Buffer placed in Posix shared memory, can be registered in parallel for InfiniBand RDMA
- Pair of ring buffers for each link
  - Data buffer for microslice data content
  - Descriptor buffer for index table and microslice meta data
Interface to Online Reconstruction Code

- Basic idea: For each timeslice, an instance of the reconstruction code...
  - ...is given direct **indexed access** to all corresponding data
  - ...uses detector-specific code to understand the **contents** of the microslices
  - ...applies **adjustments** (fine calibration) to detector time stamps if necessary
  - ...finds, **reconstructs and analyzes** the contained events

- Timeslice data management concept
  - Timeslice is self-contained
  - Calibration and configuration data distributed to all nodes
  - **No network communication** required during reconstruction and analysis