A high-throughput input interface for the CBM First-level Event Selector

Dirk Hutter
hutter@compeng.uni-frankfurt.de

Jan de Cuveland
cuveland@compeng.uni-frankfurt.de

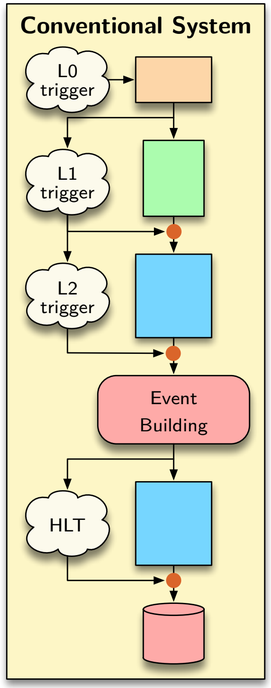
Prof. Dr. Volker Lindenstruth
FIAS Frankfurt Institute for Advanced Studies
Goethe-Universität Frankfurt am Main, Germany

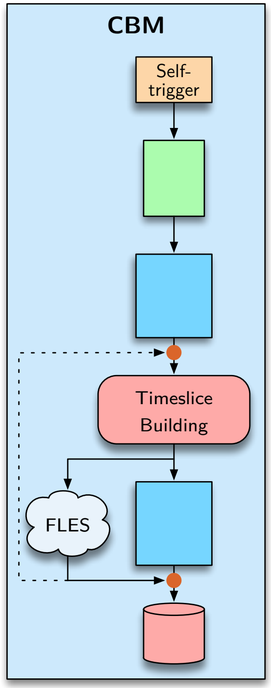
SPONSORED BY THE
Federal Ministry of Education and Research

CBM

Conference on Computing in High Energy and Nuclear Physics 2024

The Compressed Baryonic Matter (CBM) Experiment at FAIR



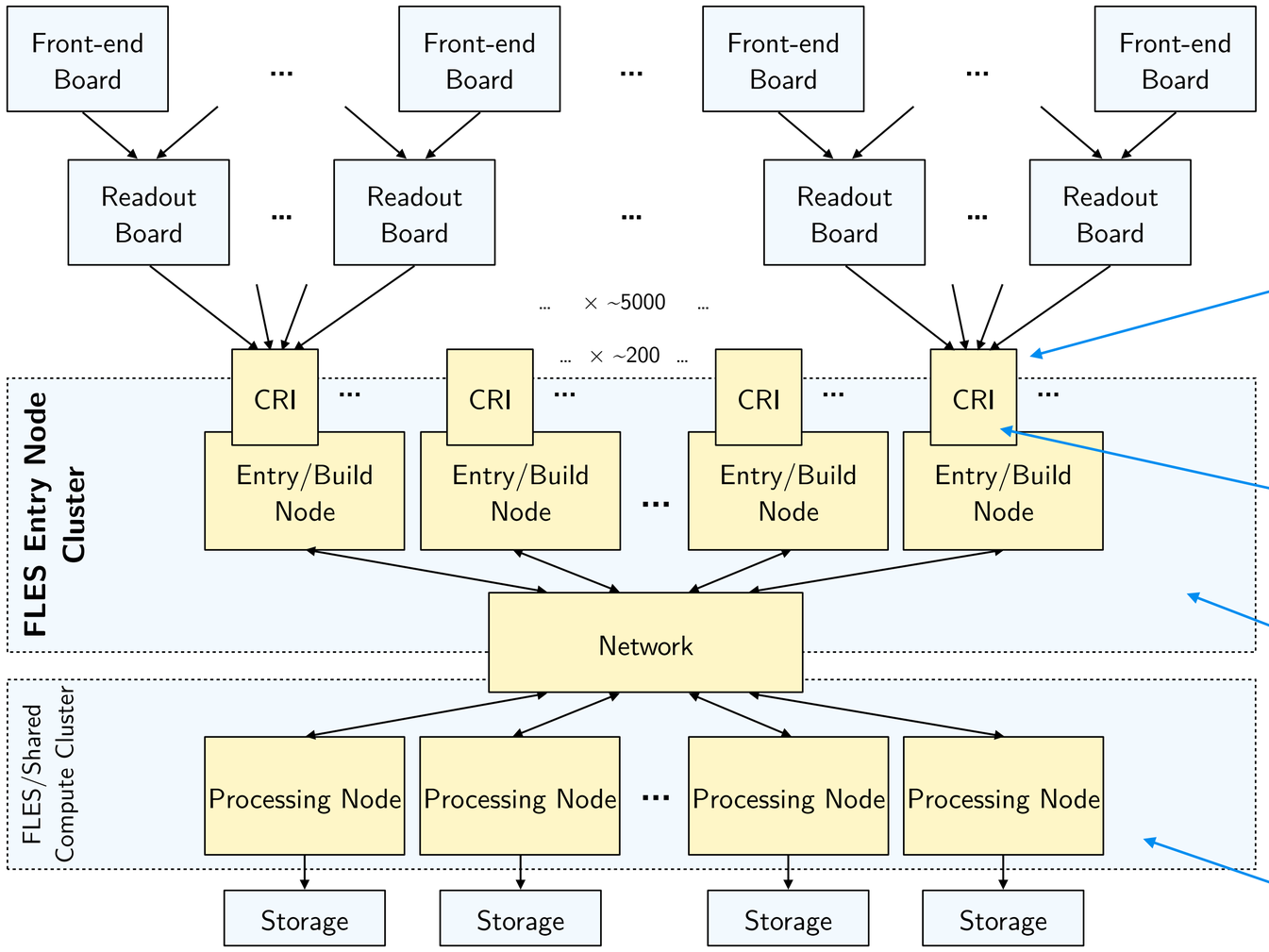


- Fixed target heavy ion experiment
- Under construction at the FAIR facility
- Up to **700** charged particles in aperture
- High reaction rates up to **10 MHz**
- Many probes lack simple trigger signatures

- **Self-triggered**, **free-streaming** readout electronics
- Event selection exclusively done in a high-performance computing cluster
- Full **online event reconstruction**

➥ First-level Event Selector (FLES)

CBM Readout and First-level Event Selector (FLES)



Self-triggered readout electronics

- Sends timestamped hit messages
- Zero-suppressed data stream

Common Readout Interface (CRI)

- FPGA-based PCIe board
- Interface to COTS nodes

FLES Interface Module (FLIM)

- HDL module as subsystem interface
- Interface between detector specific and generic DAQ data path

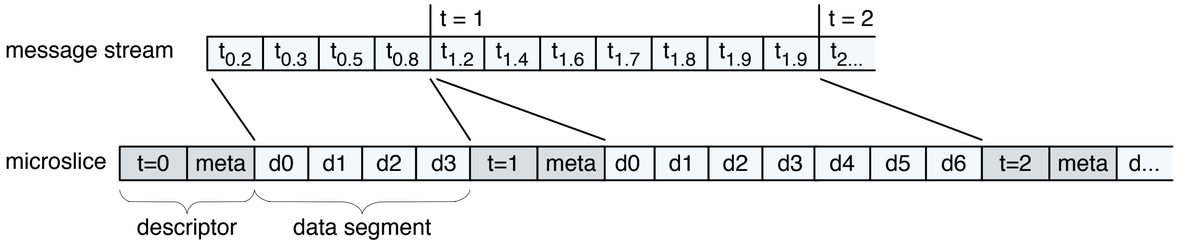
FLES Entry Cluster

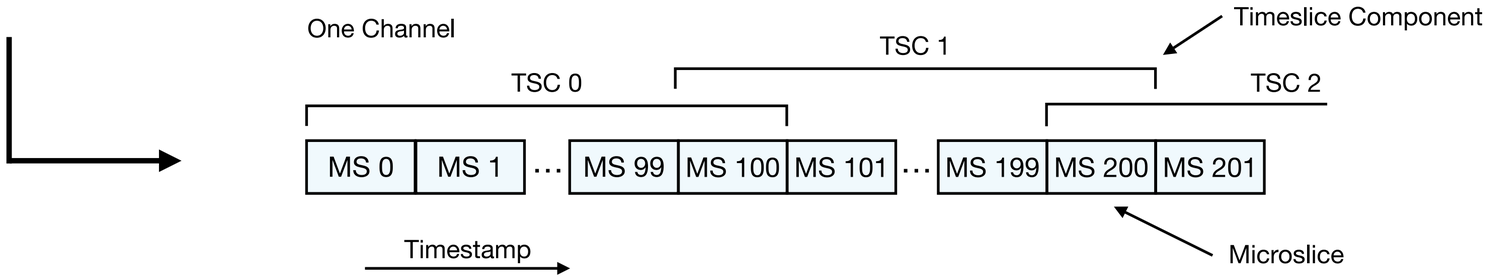
- Design data rate > 1 TB/s
- High-speed, RDMA-enabled network

FLES Compute Cluster

- Shared compute resources
- Online event reconstruction

Microslice Data Model



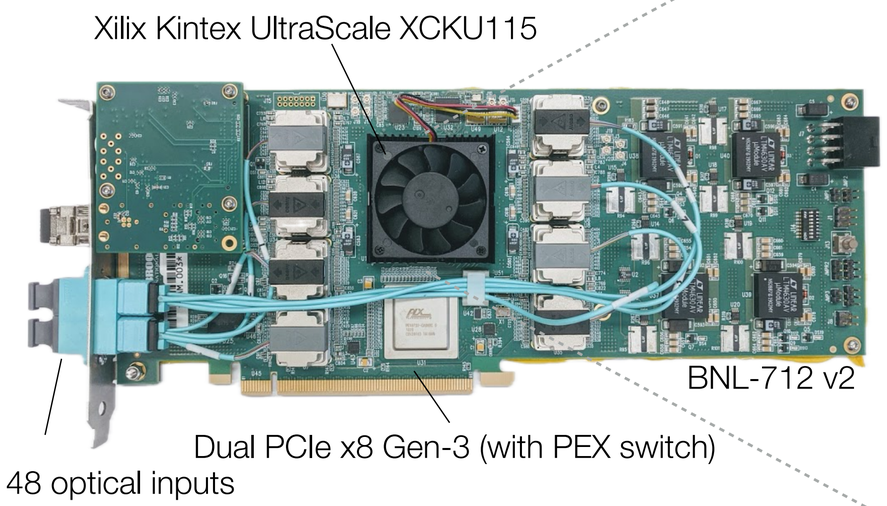


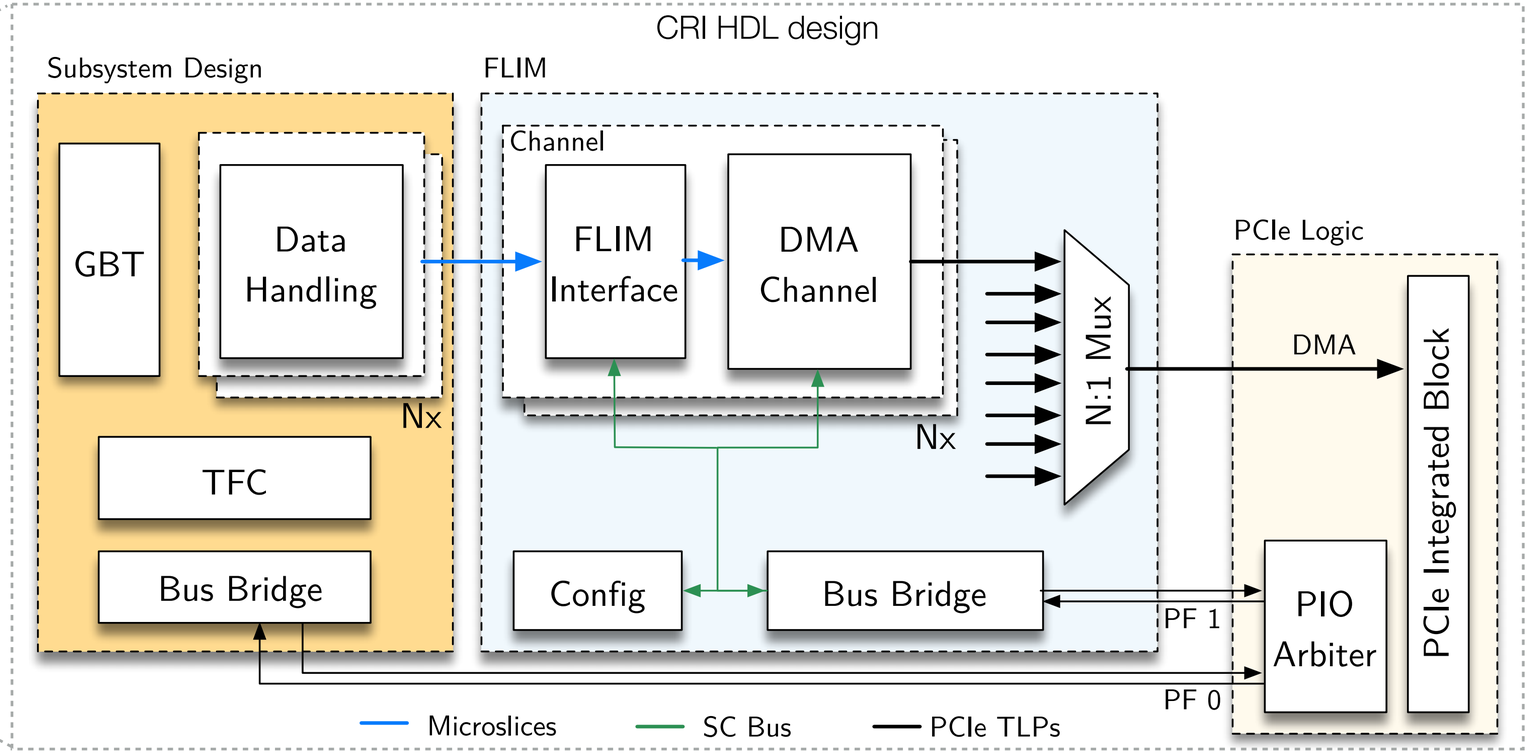
- The CRI board splits detector data streams into short, **context-free time intervals** and encapsulates them into data transport containers called **microslices**
  - Meta data provides all necessary information for data handling, e.g., **reference timestamp**
- **Timeslice building** combines subsequent microslices from all sources to processing intervals called **timeslices**
  - Microslices at the boundaries of timeslices are doubled to create **overlap**

Challenges:

- Full detector input data rate of up to **1 TByte/s**
- **Time-based** partitioning
- **Stateful streams** of heterogeneous data in **subsystem-specific** formats
- **No global event definition** at this point of the data chain
- Cut events at **boundaries**
- **Zero-suppressed data**
  -> time interval ≠ data size

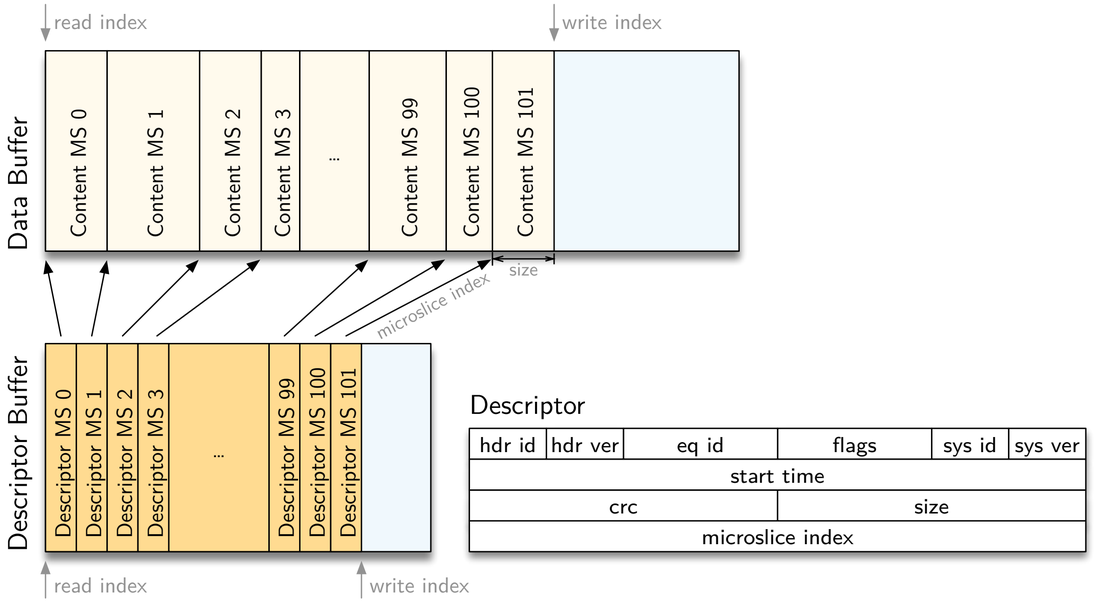
Common Readout Interface (CRI)





- FPGA-based **PCIe extension card** to the FLES entry nodes
  - Current prototype BNL-712 v2
  - Successor with Versal FPGA on the horizon
- Subsystem specific FEE interface and microslice creation
- Common **FLES Interface Module** (FLIM)
  - Microslice stream interface
  - Focus on throughput and resilience against erroneous input data
  - Scaleable to multi channel and multi PCIe
- Segregation of software stacks via PCIe physical functions (PF)

DMA Engine and Buffer Management



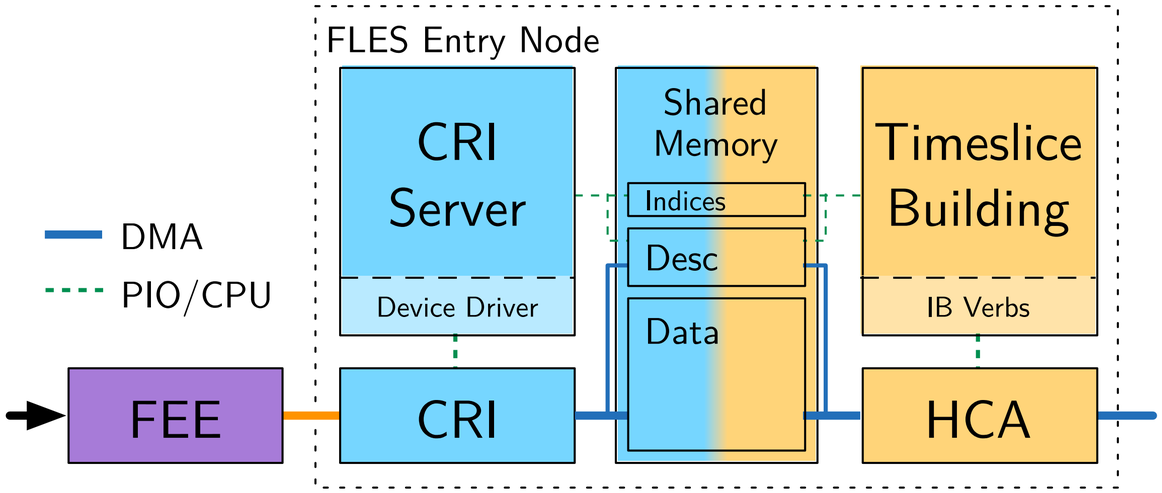
Design Challenges

- Producer–consumer problem
- High-throughput application requires DMA and asynchronous transfers
- Microslices are based on time intervals, data sizes vary widely
- Low overhead synchronization
- Avoiding memory fragmentation

- **Full-offload, scatter/gather DMA** engine
  - Multi-channel support with dynamically shared PCIe bandwidth
  - Pre-configured scatter lists remove the need for extensive data buffers on the card
- **Dual ring buffer** memory scheme
  - Data buffer holds the microslice content in a continuous stream
  - Descriptor buffer holds the fixed-sized microslice descriptors

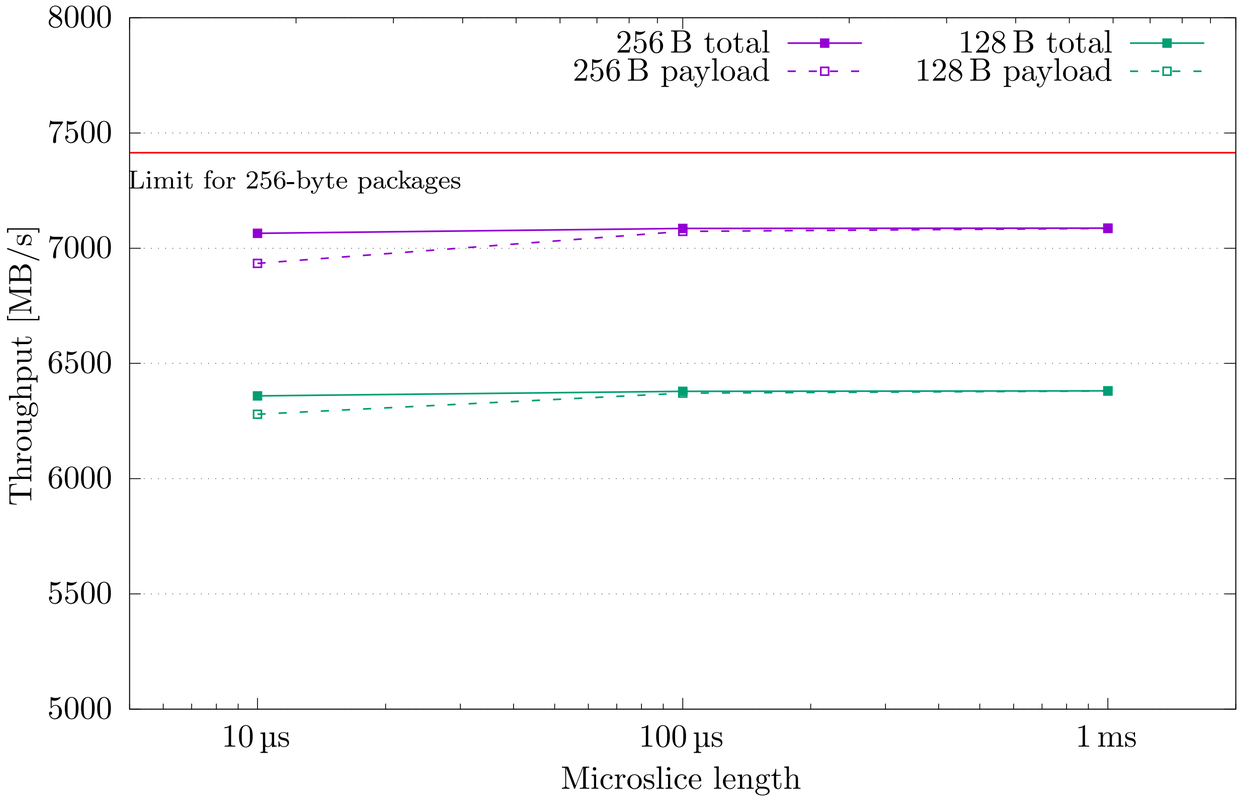
- Descriptor buffer serves as an **index table** to the microslice content
  - Monotonous index enables reuse of table after copying the data
- Synchronization via **read and write index** exchange
- Efficient, **block-wise data access**
  - Maximum four gather entries per timeslice component
  - Only two descriptor reads per timeslice component

Software Consumer Interface



- **Zero-copy** interface to timeslice building
- Microdriver architecture device driver (PDA)
  - IOMMU support for compact S/G lists
- DMA directly to **POSIX shared memory** user space buffers
  - Multi-gigabyte buffers without issues
- **DMA buffers are shareable** with other PCIe devices, e.g. an InfiniBand HCA
- Data publishing agent: **CRI server**
  - Handles all communication with the hardware and serializes requests
  - Lightweight synchronization via read and write indices
  - Multi-consumer support

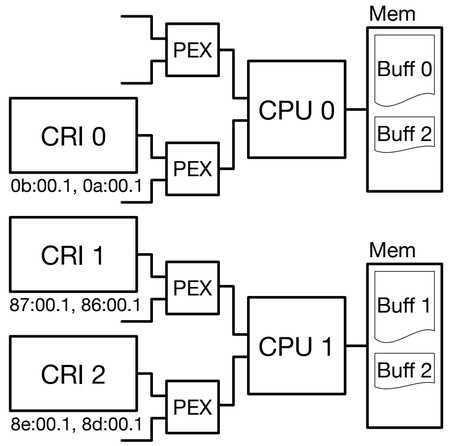
System Throughput Benchmark

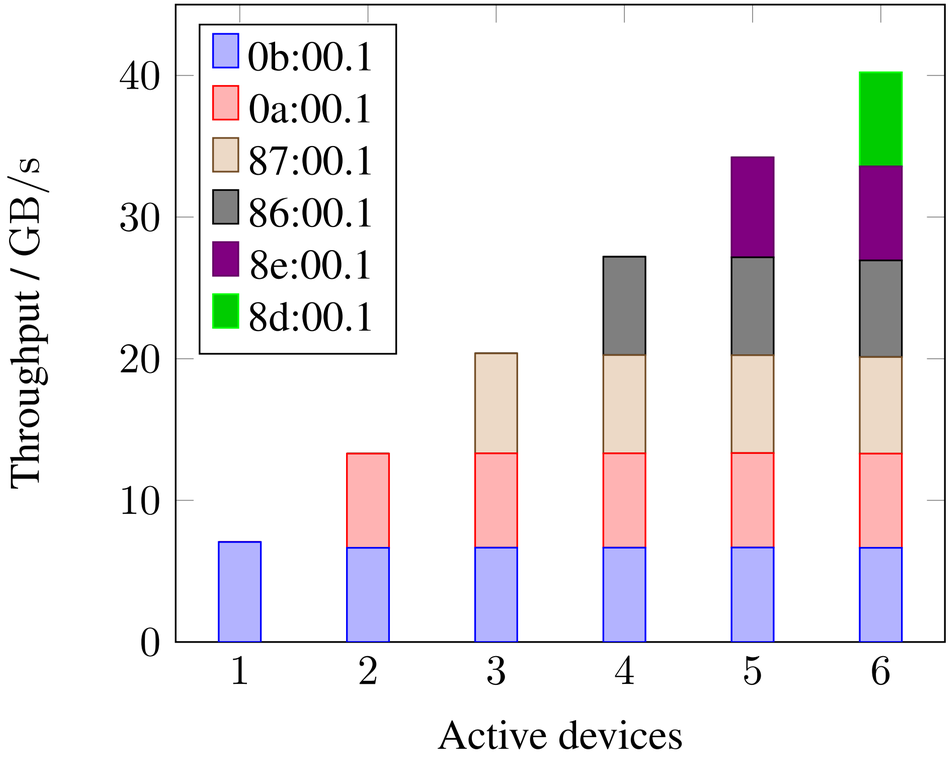


- PCIe throughput measured with internal pattern generators
  - Single PCIe 3.0 8x interface
  - Realistic, time-based microslice creation
  - Eight parallel microslice streams to oversaturate interface
- Very good performance even for small microslice sizes
  - Expected performance increase for higher PCIe payload sizes
- Full-chain throughput reaches 95% of the theoretical PCIe limit
  - Can be increased to **98% for 512 Byte packet sizes**
  - **Minimal CPU** load
  - Only limited by lack of credits from the host system

Multi-device Scaling

- FLES entry stage design foresees multiple CRIs per node
  - Simultaneous operation of multiple high-speed PCIe devices in a non-uniform memory access (NUMA) environment
- Test Setup:
  - Dual Xeon E5-2650 v4,128 GB 2400 MHz DDR4
  - 3 CRI with two PCIe x8 Gen-3 endpoints each
- NUMA aware buffer placement
  - Two of the cards write to their local NUMA node
  - The third card writes balances to both NUMA nodes
- Very promising result in small setup
  - > 7 GB/s for a single endpoint
  - 6.7 GB/s for both endpoints (limitation of the root port?)
  - Perfect scaling to multiple cards: **40.2 GB/s total**
  - **Perfect fairness**

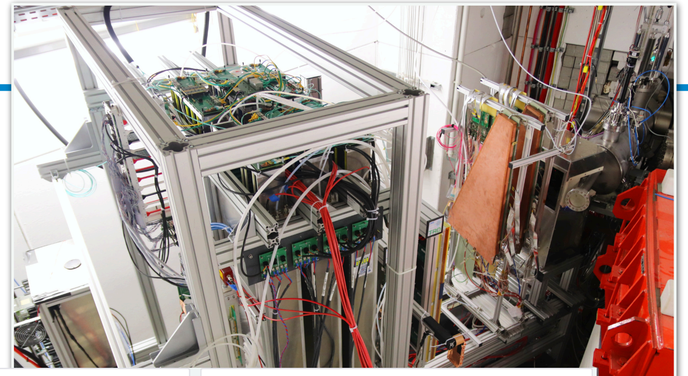


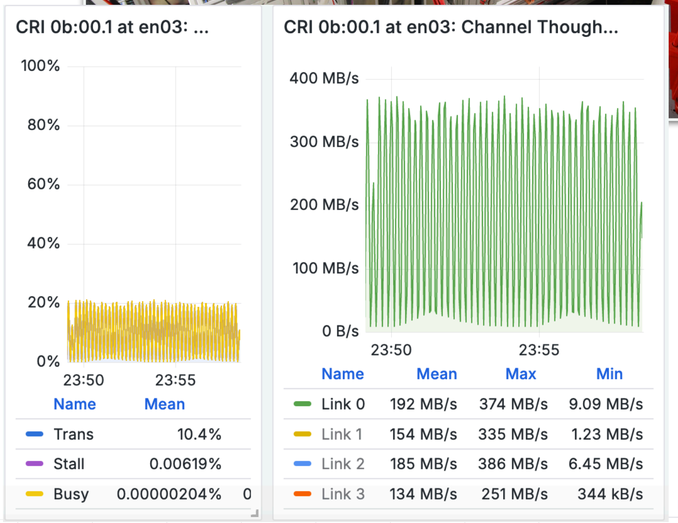


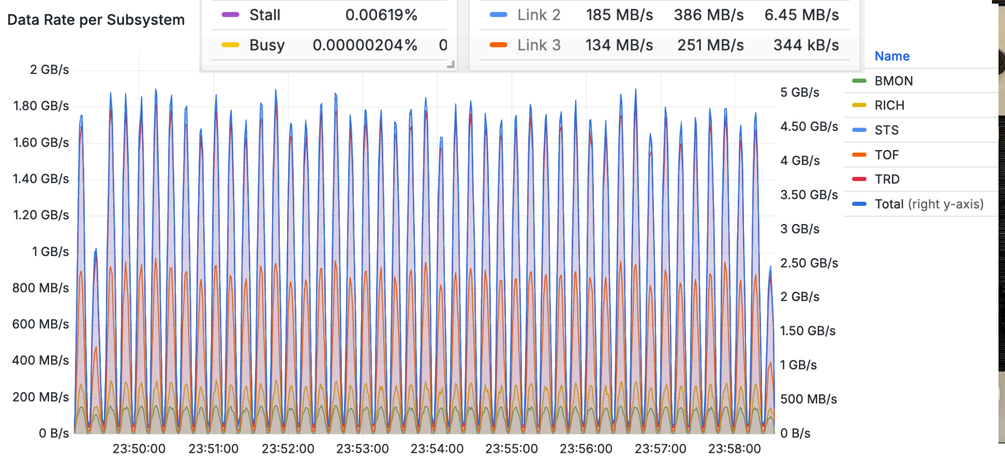
Full-system test mCBM

- The FLES input interface is in continuous use in numerous physics and development setups
- Example FAIR Phase-0 experiment mCBM
  - CRI with FLIM and FLES SW is the **central data taking system**
  - 12 CRIs in **six FLES entry nodes**
  - Regular data taking campaigns
- Control and monitoring system
  - **Hardware monitoring** of all critical parameters, e.g., microslice production rate, PCIe throughput, …
- Example 2024 data taking
  - 7 detector systems: STS, TOF, RICH, TRD, TRD-2D, MUCH and BMON
  - **6 nodes**, 8 CRIs, **57 FLIM** channels in parallel
  - Peak data rates over 5 **GByte/s**

mCBM detector setup







FLES entry stage

Run 3035 on 2024-05-10

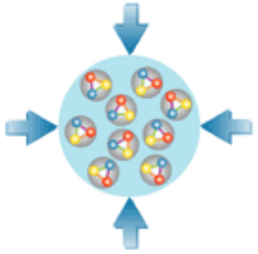
Summary

- Solution to structure heterogeneous input data: microslice data model
  - Time-based containers allow subsystem-agnostic, highly efficient data handling
- FLES interface module:
  - Provides the connectivity to the detector systems and handles all input data
- Custom DMA engine and dual ring buffer memory model for most efficient data access
  - Optimized host interface for zero-copy, high-throughput data transport
- In active use in several CBM test setups, in-beam tests and FAIR Phases-0 experiments

Dirk Hutter
hutter@compeng.uni-frankfurt.de

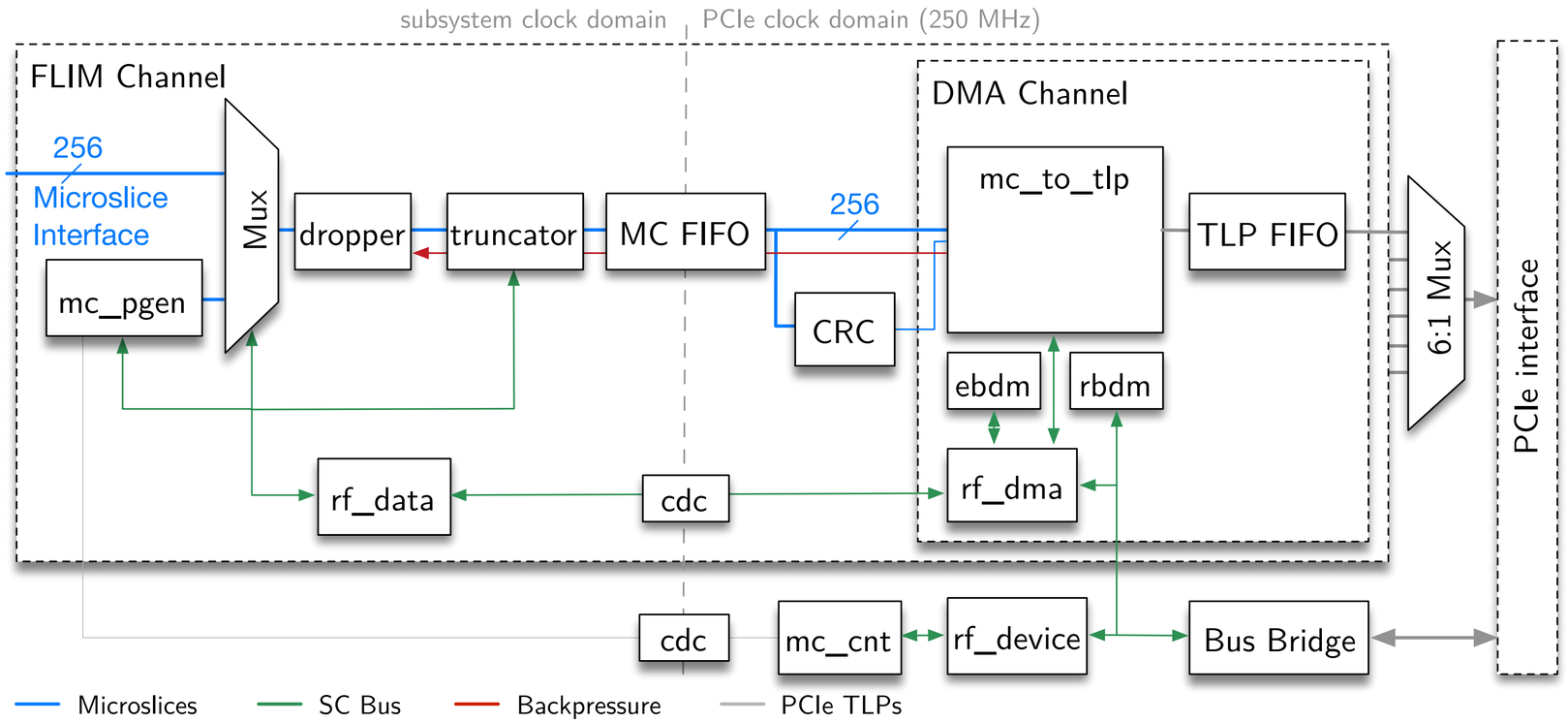
Jan de Cuveland
cuveland@compeng.uni-frankfurt.de

SPONSORED BY THE
Federal Ministry
of Education
and Research



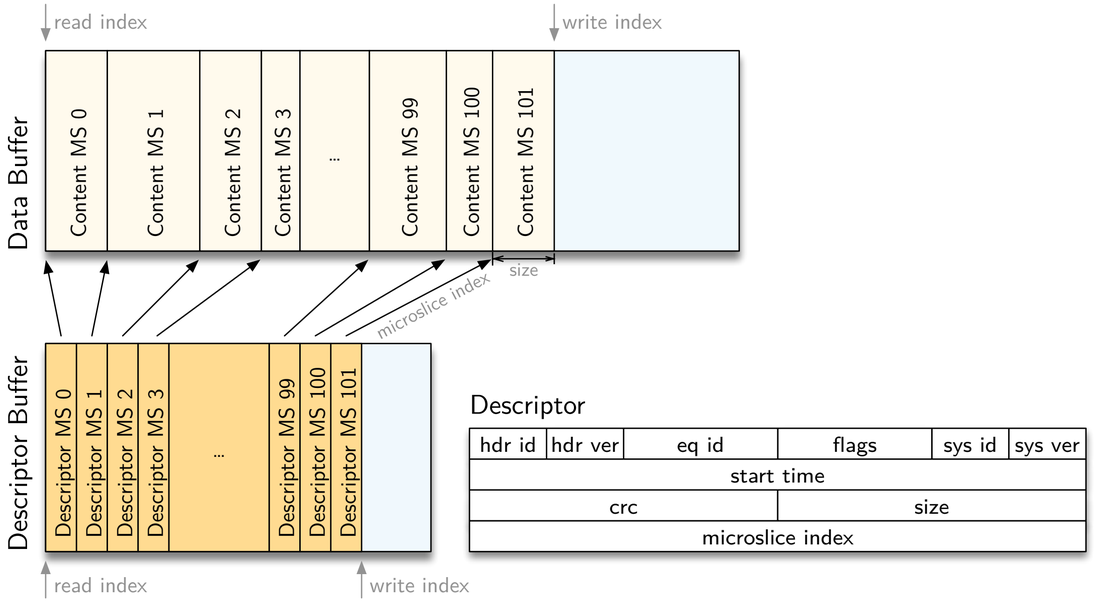
FIN

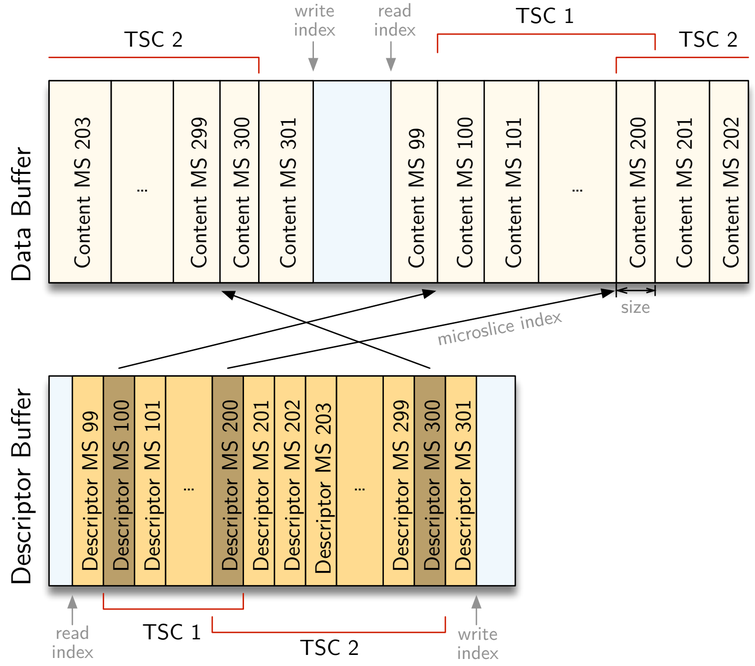
FLIM Hardwaredesign



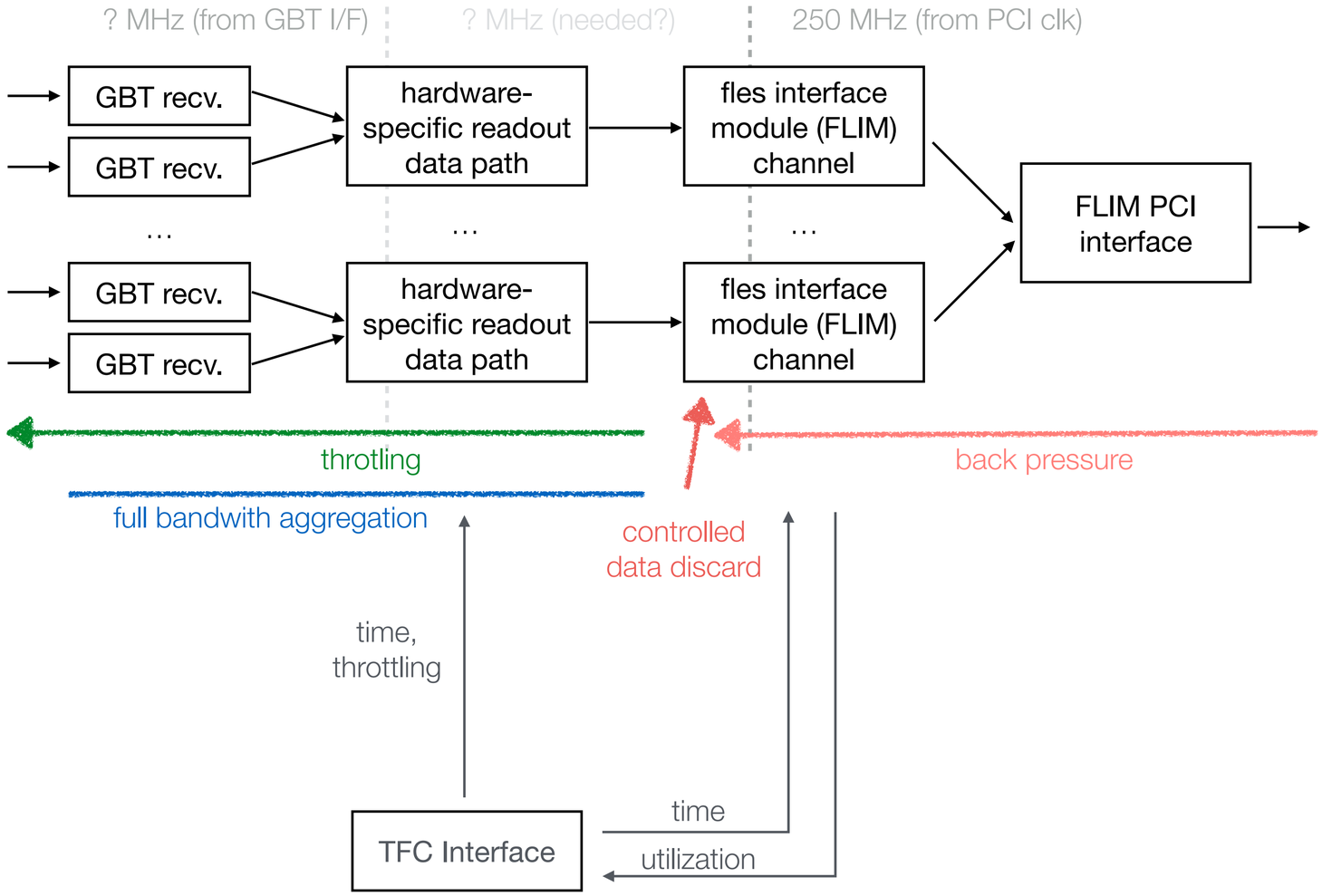
- Microslice interface:
  - **256 Bit**, variable subsystem clock (up to 250 MHz; timing target: **160 MHz**, 41 Gbit/s)
  - **Backpressure free** (may still have handshaking signals), internal drop point and derandomization buffer
  - Signals on internal **buffer fill state** (for optimizations and global throttling)
- FLIM: **6 channels** (2x 3 channels for twofold designs)

FLIM Host Interface





Proposed CRI read-out data path



Not shown here: controls, second SLR

Time Stamp Considerations (1) – Single Measurement

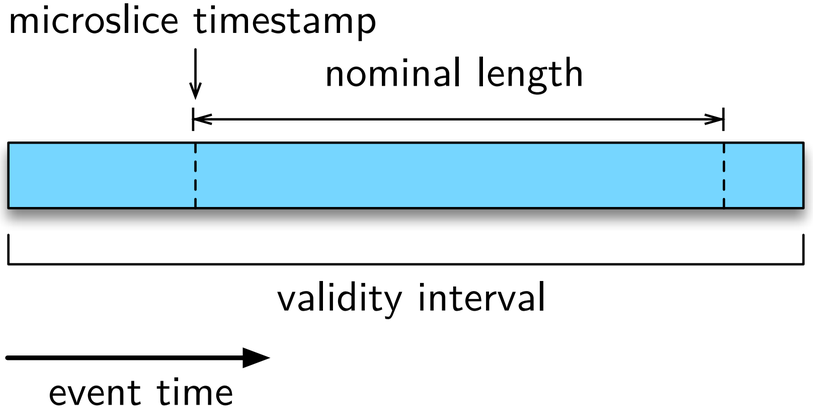
- Each measurement has:
  - A **time stamp** (known during readout)
  - The time of the associated **physics event** (assigned after reconstruction)
- These **time stamps differ** for several reasons
  - Limited precision in TFC time distribution
  - Limited precision in per-subsystem time distribution to FEE
  - Limited intrinsic detector time resolution
  - Physics and detector effects (e.g., particle time of flight, drift velocity)
- Handled by **microslice** concept
  - Specify time **uncertainty interval** of measurements (per FLES input)
  - Generate timeslices with sufficient **overlap**

Time Stamp Considerations (2) – Stream of Measurements

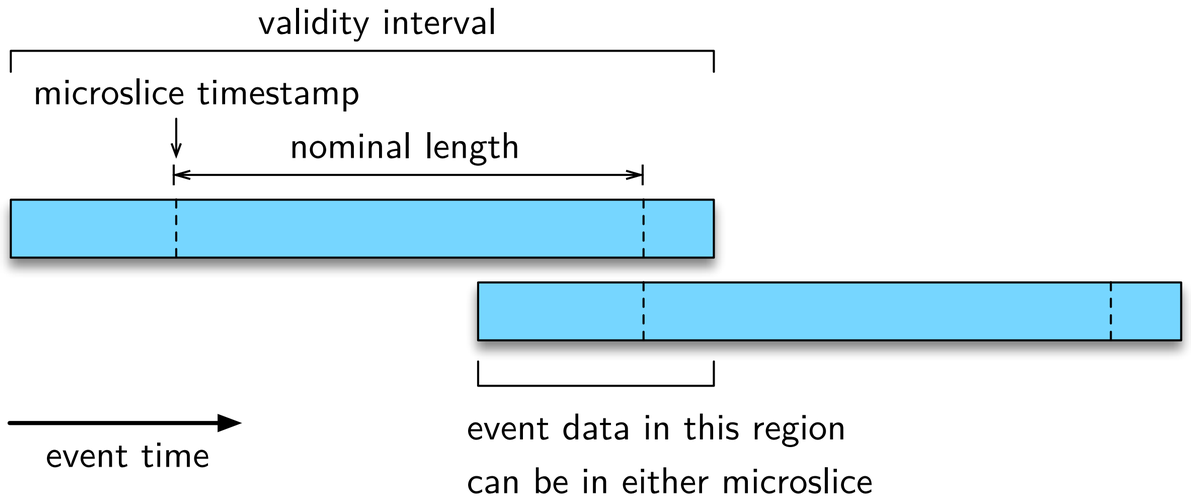
- Frontend message streams are **not** automatically merged **in chronological order** by detector readout electronics
  - Streams cannot easily be cut into intervals
- However, the maximum time deviation is usually known (e.g. unsorted only within one epoch, FEE drain time)
- We can handle this **limited time deviation** via same overlap concept
- Reduces requirements on microslice building in hardware
- CRI design implementation example:
  - Start new microslice when first measurement in corresponding time interval is encountered
  - Put any subsequent measurements into the new microslice, even if timestamp is lower again
  - Generally specify larger interval of possible corresponding event time for all microslices

Microslice Requirements

- Each component defines the maximum time deviation of **physical event data** in a microslice with respect to the microslice reference **timestamp**
  - This includes differences between message timestamp and event time, and
  - The deviation in assigning detector messages to the "nominal" microslice
- Microslice guarantee: All included measurements have an **event time in the validity interval**
  - Note: this doesn't imply that all measurement with event time in the validity interval are included

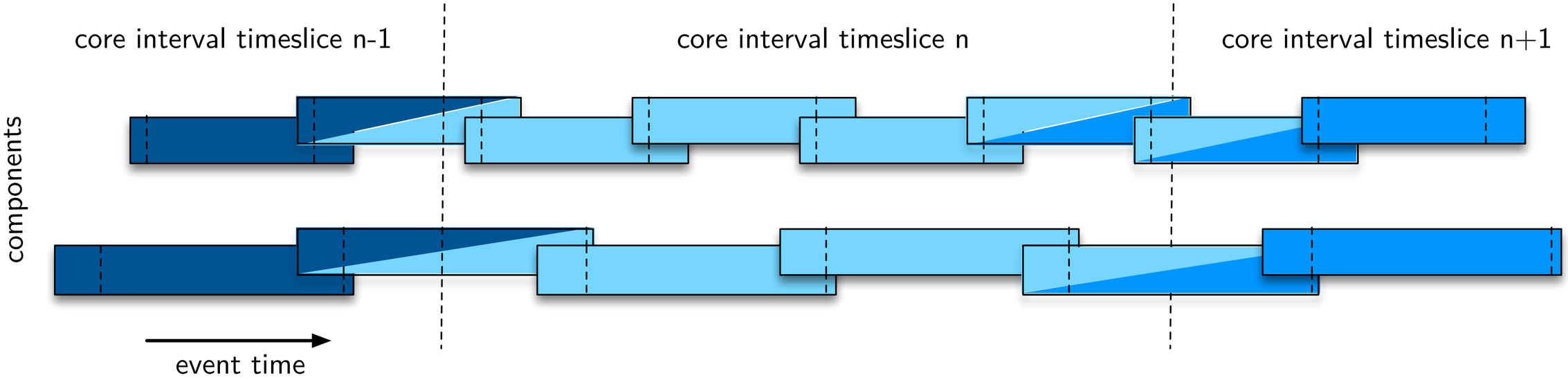


Microslice Building Design Guidelines



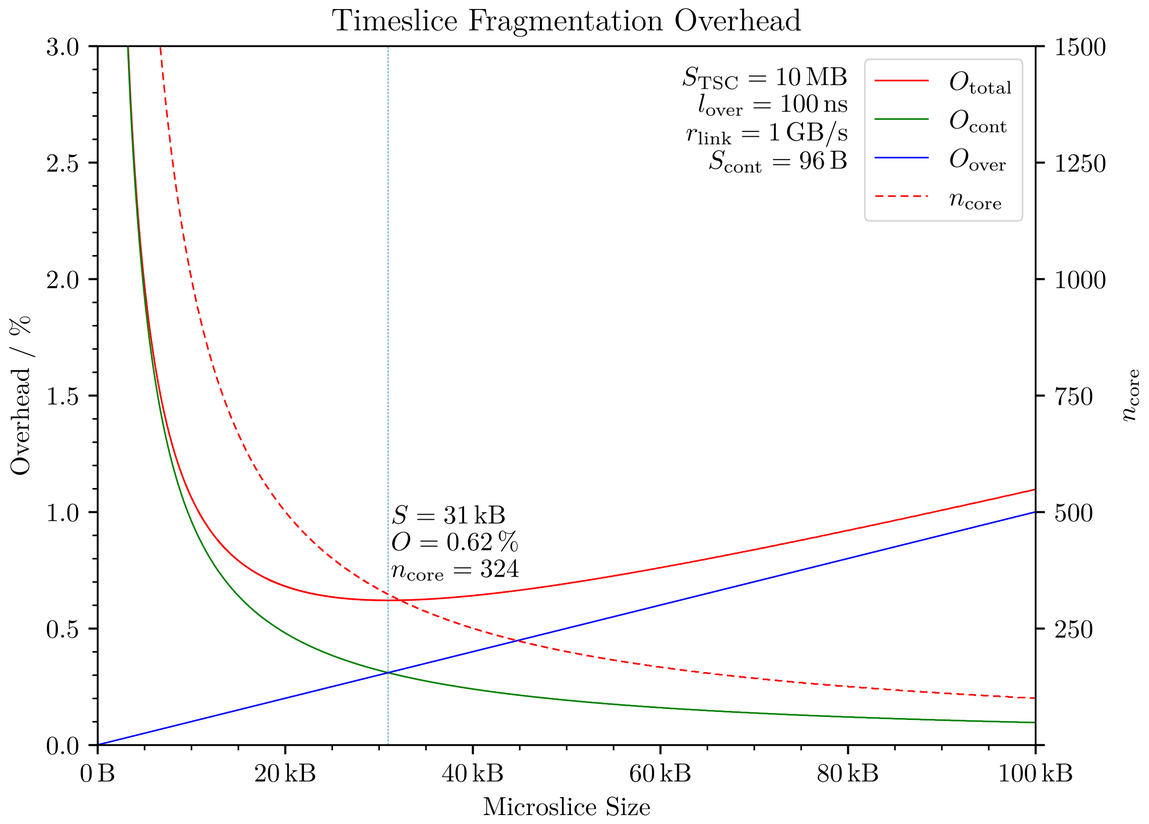
- The nominal microslice length and validity interval length are fixed for each subsystem (or component)
  - Future plans: Do not require a globally common microslice length.
- No sorting required within the microslice, software has to sort the messages anyway
- The nominal intervals between multiple components align in event time
  - Microslice timestamp modulo nominal length = 0 if feasible
- Note: this definition is fully backwards compatible with the initial requirements

Timeslice (Component) Building



- A timeslice is the collection of all microslices with a validity interval intersecting the timeslice core interval.
- Timeslices overlap, but without an explicitly defined overlap region
- Timeslice guarantee: All measurements with **event time in core interval** are included

Scaling and Packaging Efficiency



- CBM has a wide range of running scenarios
  - FLES is expeced to work with a fraction of compute and network resources in the start version
- Microslices allow to scale the system to lower rates by increasing the length of a microslice
  - Keeps the overhead proportional to the data rate
- Optimal microslice size can be calculated from overhead for a given target configuration
- Overhead components
  - Overhead from packaging of microslices
  - Overhead from duplication of microslices for overlap region