Data preprocessing of the DAQ system for TOF detector in CBM experiment

Wenxiong Zhou, Pierre Loizeau, Jochen Frühauf, Junfeng Yang, David Emschermann, Walter F.J. Müller for the CBM collaboration

GSI Darmstadt
<zhouwenxiong@cqu.edu.cn>
Outline

- Introduction of CBM experiment
- Structure of DAQ for TOF subsystem
- Data preprocessing strategy
- Outlook
1. Introduction of CBM experiment
1. Introduction of CBM experiment

Facility for Anti-Proton and Ion Research

SIS100 beam energy
10 AGeV/c Au-beam

FAIR

Facility for Anti-Proton and Ion Research
1. Introduction of CBM experiment

A fixed target, high interaction rate, heavy ion physics experiment

Data pre-processing FLES input cluster

Online timeslice building and event selection

μ-slices

1 TB/s

~700m

~80m

Data flow
1. Introduction of CBM experiment

Ref: HK 6.1 for TOF Detector

CBM

- Dipole Magnet
- Silicon Tracking System
- Micro Vertex Detector
- Transition Radiation Detector
- Ring Imaging Cherenkov
- Time of Flight
- EMCal
- Projectile Spectator Detector
- Muon Detector
2. Structure of DAQ for TOF subsystem
2. Structure of DAQ for TOF

Conventional DAQ system:
1. Multi-Level trigger system. (several microseconds are needed for level 1 trigger)
2. Pipeline readout.

2. Structure of DAQ for TOF

**CBM requirement:**
Complex algorithms are needed for trigger.

**Features:**
1. Data must have timestamp to allow event selection in computer farm.
2. High transfer speed is needed to acquire all the data.


2. Structure of DAQ for TOF

**DPB**: Data Preprocessing Board.
**FLES**: First Level Event Selection.
**FLIB**: FLES Interface Board.
**GET4**: FEE for TOF detector. There are 4 channels for one chip.

**Epoch Message**: timestamp from GET4.
**Hit Message**: data information from detector sent out by GET4.
**MicroSlice(MS)**: data frame used in DPB board.
**TimeSlice**: data frame used in computer farm.
2. Structure of DAQ for TOF

The diagram illustrates the logic of the clock, epoch, and sync. It shows:

- **Sync**: A single pulse lasting about 0.83 seconds.
- **Epoch**: A series of pulses with a continuous sequence of hits and errors, indicating a floating length.
- **CLK**: A series of regular pulses with a defined length of 25.6 microseconds.

The structure of MicroSlice is depicted with the following details:

- **Epoch**: Starting point.
- **Hit Hit Error Hit Hit Hit Hit**: The sequence of hits and errors within the epoch.
- **Epoch**: Ending point, indicating the completion of the structure.

The diagram provides a clear visualization of the timing and sequence of events in the DAQ system.
2. Structure of DAQ for TOF

Current structure of the DAQ used for STAR testing system.
2. Structure of DAQ for TOF

GET4 Board
(8 GET4 chips/board)

AFCK
DPB Board

DAQ PC
FLIB
2. Structure of DAQ for TOF

Ref: HK 20.5 for GBTx
3. Data preprocessing strategy
3. Data preprocessing strategy

![Diagram of data preprocessing strategy with labels:
- FPGA
- GET4 Interface
- Ethernet
- Clock & Sync
- 10Gbit/s Optical Link]
3. Data preprocessing strategy

**FLIM**: FLES Interface Module
3. Data preprocessing strategy

- **RealMicroSlice (RealMS)** basic requirements
  1. All GET4 share one same epoch.
  2. Hit message must be sorted at first (to be implemented).
3. Data preprocessing strategy

- **Requirements**

1. Data acquisition has to detect and remove all different kinds of error:
   - **Epoch loss**: caused by GET4 errors.
   - **GET4 stop sending data**: because of GET4 broken.
   - **Corrupted data**: due to interference.

2. Broken GET4 should be recovered if it is possible.
3. Data preprocessing strategy

One possible strategy

<table>
<thead>
<tr>
<th>Input speed</th>
<th>Channel</th>
<th>Read speed</th>
<th>Read clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>80Mbit/s</td>
<td>&lt;=32</td>
<td>2-3 cycle/ data</td>
<td>&gt; 160MHz</td>
</tr>
</tbody>
</table>
3. Data preprocessing strategy

Improved Data Merging strategy (It is used now)

<table>
<thead>
<tr>
<th>Input speed</th>
<th>Read speed</th>
<th>Merging speed</th>
<th>Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>80Mbit/s</td>
<td>2 cycle/data</td>
<td>1 cycle/data</td>
<td>64</td>
</tr>
</tbody>
</table>
3. Data preprocessing strategy

Error detecting and removing strategy

Error detection:
1. There are no epoch message in one GET4 for 30μs: epoch interval time is 25.6 μs, and there may be not epoch sent out from GET4 due to chip broken.
2. There are mismatch for epoch: the received epoch counter number is different from the counter number got from local FPGA.
3. Same epochs are received from one GET4: a broken GET4 may send same epoch for several times.
4. Epochs are lost: the received epoch counter number is not continued.

Error processing:
1. Synchronization command are sent out for Error GET4 chips every 2s (It is also a method used to recover GET4 chips).
2. Remove the same epoch.
3. Repair the lost epoch.
3. Data preprocessing strategy

Real MS

1. All epochs are aggregated into one epoch with a special ID.
2. Data length is not fixed.
3. The interval time between two epoch marks is about 25us.
3. Data preprocessing strategy

Test setup for the preprocessing firmware
3. Data preprocessing strategy

**S.C.M:** Slow Control Message
3. Data preprocessing strategy

![Data preprocessing strategy diagram](image)
3. Data preprocessing strategy

DAQ system testing in STAR

μTCA crate with AFCKs
4. Outlook
4. Outlook

1) Readout chain based on GBTx

Ref: HK 20.5 for GBTx
4. Outlook

2) Real-time data sorting

Reason to sort:
1. Epoch message are sorted by GET4
2. The sequence of Hit message is disrupted when the state machine of GET4 read data from different channels.

4. Outlook

3) DAQ based on CRI

\[ \text{CRI} = \text{AFCK} + \text{FLIB} \]

Ref: HK 20.5 for CRI
Thank you!
Backup

- SimplifiedMS
  - 1. Each GET4 has their own epoch event.
  - 2. Epoch event from GET4 are used directly.
- RealMS
  - 1. All GET4 share one same epoch.
  - 2. Hit message must be sorted at first.
• Time is the only way to organize data at DAQ level
• Events {'one interaction'} are defined during reconstruction

→ Time slice building instead of 'event building'

• Requirements for time slice building
  • collect all data of a time slice in one compute node
  • adjacent time slices must have some small overlap
    {otherwise interactions at the slice boundary can not be reconstructed}

• Efficient solution
  • define a micro slice, length few μsec
  • macro slice is build from micro slices
    {e.g. 100 mirco slices with 1 micro slice overlap}
Backup

Data-Stream-in

Time Comparator (first level discrimination)

- Time 4-5us or 4-5us of (t%5)
- Time 3-4us or 3-4us of (t%5)
- Time 2-3us or 2-3us of (t%5)
- Time 1-2us or 1-2us of (t%5)
- Time 0-1us or 0-1us of (t%5)

- Main Buffer Memory-4
- Main Buffer Memory-3
- Main Buffer Memory-2
- Main Buffer Memory-1
- Main Buffer Memory-0

- Dynamic Memory
- Next Data availability bit array
- Data availability status bit array
- Data Status bits
- Next Address array associated to Main memory
- Next data valid bit arrays associated with both Main and dynamic memory
- Next address array associated to dynamic memory

Sequencer

Date-stream-out

Address list array 1
Address list array 2
Address list array 3
Address list array 4
Address list array 5

Zero suppression Block