MPD Data Acquisition System

Technical Design Report

Version 2018.8.27

MPD DAQ Collaboration

A. Baskakov, S. Bazylev, A. Fediunin, I. Filippov, S. Kuklin, A. Shchipunov, A. Shutov, I. Slepnev, V. Slepnev, N. Tarasov, A. Terletskiy

Laboratory of High Energy Physics
Joint Institute for Nuclear Research
Dubna, RUSSIA
27 August, 2018

Abstract

This part of the MPD Technical Design Report describes the Data Acquisition System (DAQ). The core function of the DAQ system is realization of data transfer from the detector to the storage system. It includes the data flow from readout electronics to the First Level Processor (FLP) fabric, to the Event Building (EB), High Level Trigger (HLT) and to the Storage System. Main DAQ components are data transfer networks, data processing servers, online storage system, software packages, network communication protocols and data formats. Readout electronics interface, Clock and time synchronization (Timing) System, Trigger System are also covered in this report.

Contents

1 Introduction

A conceptual design of the Multi Purpose Detector (MPD) [1] is proposed for a study of hot and dense baryonic matter in collisions of heavy ions over the atomic mass range A=1–197 at a centre-of-mass energy up to $\sqrt{S_{NN}} = 11GeV$ (for Au_{79+}). The MPD experiment is foreseen to be carried out at a future JINR accelerator complex facility for heavy ions – the Nuclotron-based Ion Collider fAcility (NICA) which is designed to reach the required parameters with an average luminosity of $L = 10^{27}cm^{-2}s^{-1}$

The MPD experimental program incorporates a wide variety of running conditions. A large number of trigger logic setups will be used to select events relevant to physics topics being studied. The amount of data taken in experimental runs varies with the trigger conditions. The most challenging in terms of data amount are the heavy ion collision runs. They drive the requirements on DAQ performance, network bandwidth and storage capacity.

The selection of events in the MPD experiment is performed by the Trigger System. The Level 1 trigger Processor (L1TP) and Central Trigger Processor (CTP) are hardware based. Logical level 0 trigger signals are received by L1TP, aligned by delays and processed by hardware trigger logic to produce Level 1 Trigger. CTP is driven by L1 trigger and controls the L2 trigger sent to DAQ. The High Level Trigger (HTL) is run in software on events received and built by DAQ, its purpose is to select events to be archived to permanent storage.

2 System Requirements

2.1 Modes of Operation

These modes of DAQ operation are defined:

- DAQ only, HLT (High Level Trigger) disabled

  This mode should be available at any time of MPD run. It is the initial point of DAQ operation. No data is sent to HLT.

- DAQ + HLT analysis

  The HTL is active, but not enabled for trigger. It may not remove or modify data, however it may add data to the events. The purpose of this mode is to enable the possibility of verifying the HLT algorithms.

- DAQ + HLT enabled

  The HLT is enabled and fully functional. It may take trigger decision to accept or reject the events.

2.2 Partitioning

Partitioning is the capability of controlling and building events from part of the system independently and concurrently with the rest of the system. If partitions will use same types of triggers then partitioning become splitting Event Building system. If partitions will use triggers of different types, then there will be several Run Control programs which will control its own part of system and Run Control Supervisor.

2.3 Performance Requirements

MPD will study several physics topics using different beam conditions. A large number of trigger classes are used to select and characterize the events. These trigger classes belong to two broad categories depending on whether they are frequent or rare.

Some of the trigger classes (central, semi-central and minimum-bias) are so frequent that the limiting factor is TPC maximum trigger rate. These triggers use a very large fraction of the total DAQ bandwidth. All parts of the data pipeline should allow uninterrupted flow of events from readout electronics to the Permanent Data Storage (PDS).

3 DAQ Architecture

3.1 Initial Requirements

The requirements to the DAQ system performance follow from the physics tasks. At a design luminosity of $L = 10^{27} cm^{-2} s^{-1}$ for $Au + Au$ collisions at NICA the interaction rate is about $7kHz$ [1]. The particle multiplicities are $\sim$ 1000 for the central collision at $\sqrt{S_{NN}} = 9GeV$. Thus the average sustained event rate handled by the DAQ system should be 7 kHz.

3.2 Event Size

The event size resulting from heavy-ion collisions is directly proportional to the multiplicity. It is a function of the centrality of the interaction.

Largest impact on data size give TPC and ECal subdetectors at the first stage of MPD construction. In final stage MPD will have Inner Tracker (IT) that will produce the raw data at a rate of some GiB/s. It should be planned in DAQ architecture from the very beginning.

Detector	Number of FEE Channels	Channel Occupancy	Information Type	Event Size (KiB)	Readout Rate (MiB/s)
ECal/B	45360	0.09	ADC Samples, 30 bytes	122	834
FFD	96	-	Timesatmp	-	< 100
FHCal	700	-	ADC Samples	-	< 100
TOF/B	13440	0.15	Timestamp + Width 12 bytes	24	162
TPC	95232	0.6	SAMPA Samples, encoded	875	5500
Total				$\sim$ 1000	$\sim$ 6500

3.3 Data Rate Estimation

The data from detector readout electronics boards is encoded in MPD Raw Data Format 8.1. Besides the digitized data from front-end electronics it contains extra 24-byte block of related meta information. The overhead factor is variable and for TOF TDC data formatting adds 30% extra data.

The formatting bits do not carry information necessary to reconstruct physical interaction event but are required for operation of event building software.

For estimation of the average DAQ bandwidth required for processing at design event rate the minimum bias event parameters are used. It is essential that the peak bandwidth requirement is 3 to 10 times higher, depending on the number of events that may be buffered in readout electronics before processing.

3.4 Data Flow

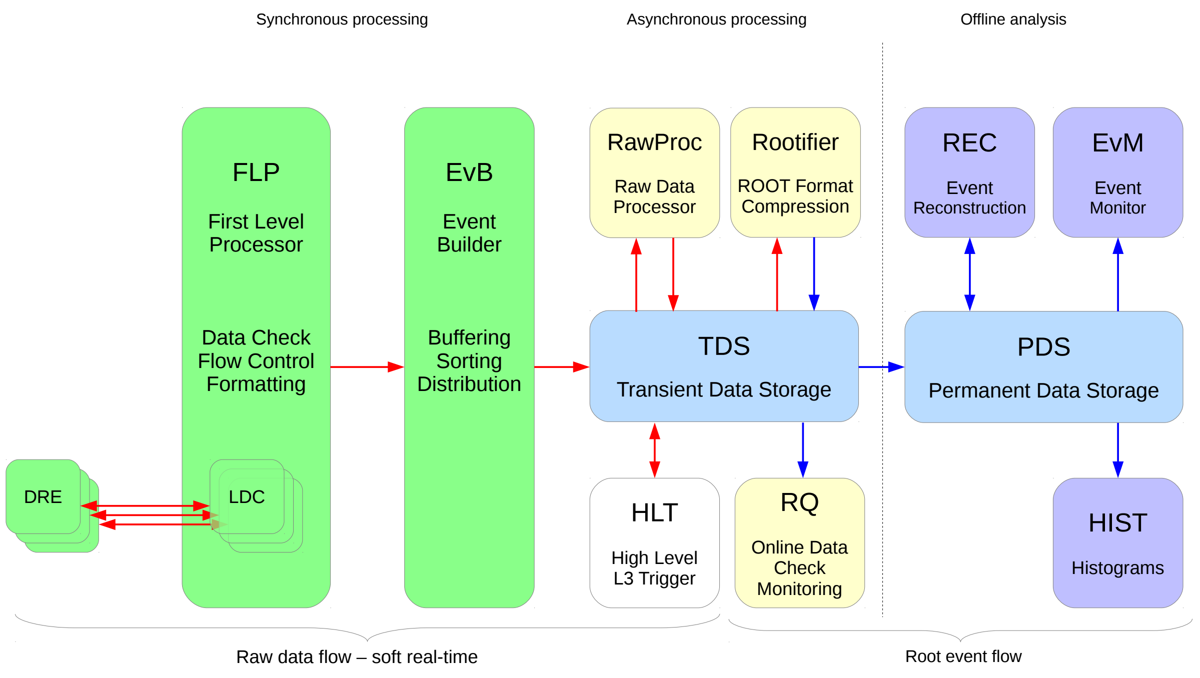


Figure 1: MPD DAQ Data Flow

The data-flow from the detector electronics up to the permanent data storage is organized as a sequential data-driven pipeline. Upon reception of a sequence of trigger signals requesting the data collection, the selected elements of the detectors generate data fragments that are transferred to common readout units and later to computers via optical links. MPD DAQ computing farm is used to check, label, format and record the data.

The MPD data processing pipeline is comprised of the following elements that are grouped into processing modes.

Synchronous processing. Real-time operations that have direct impact on data acquisition process and may throttle trigger system in case of system saturation and overflow.

1. DRE - Detector Readout Electronics, or Common Readout Unit (CRU), a data aggregation electronic boards that combine multiple low-throughput data streams into larger streams and also have interfaces to trigger, timing and clock distribution systems and front-end electronics control systems.or signal digitizer boards located inside MPD barrel

2. LDC - Local Data Collector, a software or hardware element that receive raw data from DRE (CRU). It is implemented as high throughput network adapter or as a custom designed PCI-Express board, depending on the detector readout type.

3. FLP - First Level Processor, a computer node that receive raw data stream from detector electronics, performs data validation and data fragment reassembly. MStream protocol receivers are run on FLP nodes.

4. EvB - Event Builder, a computer system that assembles data block from multiple parts into ordered sequence of data blocks that correspond to physical events when using triggered or time-slice readout

Asynchronous processing, decoupled from synchronous processing by Transient Data Storage.

1. RawProc - Raw data processor, a hardware and software system that extracts signal parameters from digitized signals.
2. Rootifier - converter from raw to compressed ROOT format. Data in ROOT format is transferred to Permanent Data Storage where offline analysis is performed.
3. HLT - High Level Trigger, a software system that main function is sorting out data corresponding to unnecessary (not important for physical analysis) events by using fast event reconstruction. Planned for MPD Stage-2.
4. RQ - Raw Data Quality Check, a continuously running program that decodes full raw data stream and performs data validation to ensure that the readout electronics and DAQ system are operating properly and all required detectors data is present in the raw stream
5. TDS - Transient Data Storage, a high throughput and low latency system that has primary goal to store raw and processed data for short time period (24 hours) to enable continuous data taking process

Offline Analysis (not covered in this TDR)

1. PDS - Permanent Data Storage, a high capacity system that provides space for all the experimental data for the whole data retention period.
2. HIST - Online Histogram web service, for displaying histograms of critical distributions to ensure that the MPD is operating correctly, runs on reconstructed data
3. REC - Data Reconstruction processes, runs asynchronously on PDS
4. EvM - Event Monitor

4 Subdetector DAQ Architecture

4.1 TPC DAQ

4.2 TDC Based Readout

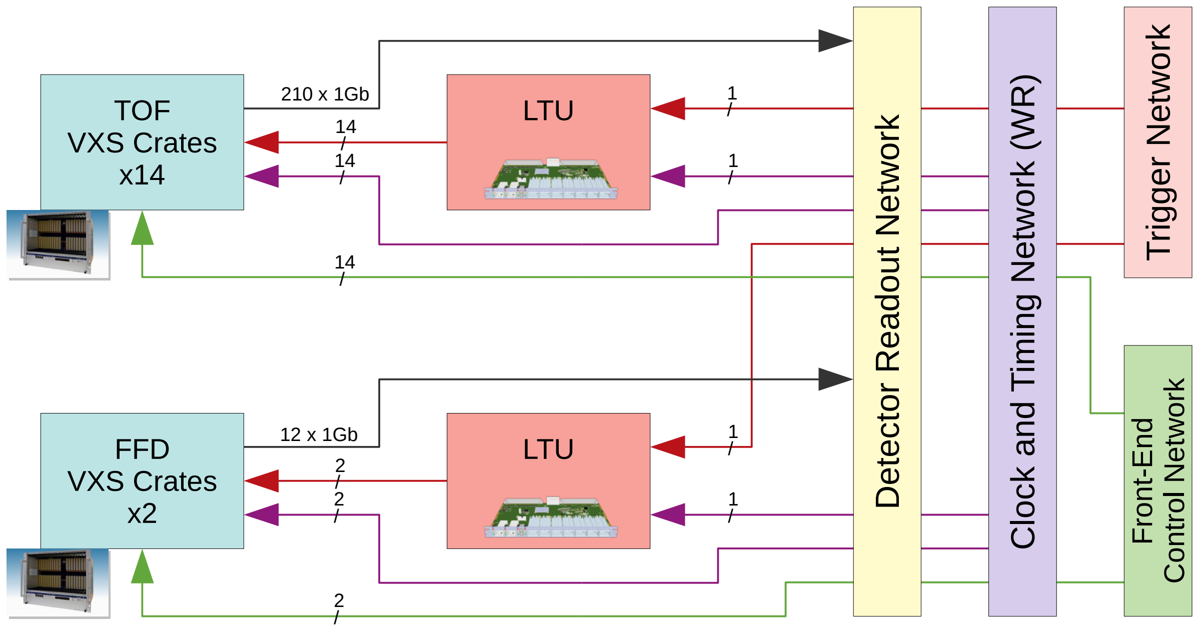


Figure 2: TDC Based Readout

TDC based readout used for TOF and FFD subdetectors. Both subdetectors uses similar DAQ architecture - Local Trigger Units (LTU) for trigger distribution and VXS crates for installing Detector Readout Electronics (DRE). Quantity of VXS crates and DREs depends on subdetectors channel numbers and geometry. Both subdetectors has 1 LTU. LTU has 1 connection to Trigger Network and 1 connection to Clock and Timing Network for synchronization.

4.2.1 TOF DAQ

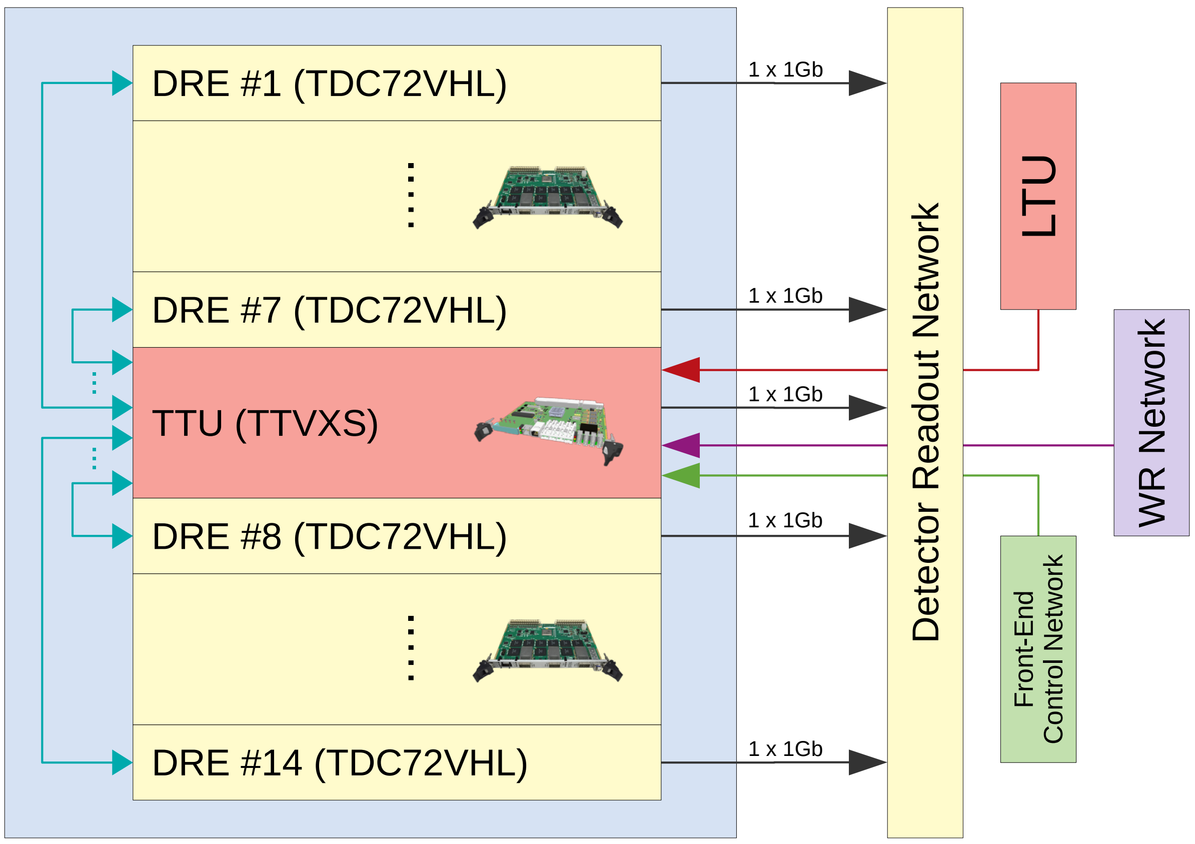


Figure 3: MPD TOF DAQ Network

There are total of 14 VXS Crates planned for TOF DAQ. In each crate will be installed 1 Time and Trigger Unit (TTU) and 14 DREs. TTU receives trigger information from its LTU, synchronization from Clock and Timing Network and controlled via Front-End Control Network. Inside VXS crate TTU distribute trigger and synchronous clock to DREs. DREs sends busy signals in reply to trigger back to TTU. Also TTU monitors status of every DRE (voltages, temperature and etc). TTU and DRE sends data via 1Gb Ethernet, 1 link per module.

Number of LTUs	1
Number of VXS Crates	14
Number of TTUs	14
Number of DREs	196
Number of data readout links	210
Data readout throughput (per link)	1 Gb
Number of White Rabbit links	15

Table 1: TOF DAQ Specification

4.2.2 FFD DAQ

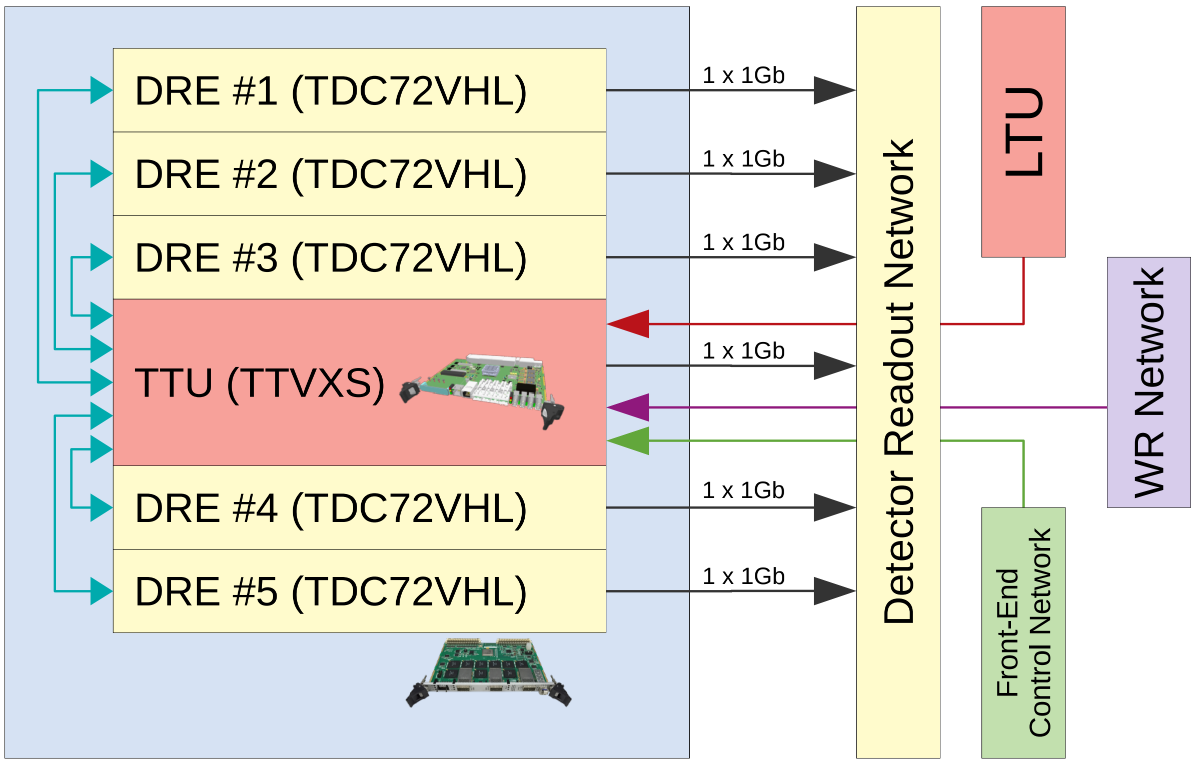


Figure 4: MPD FFD DAQ Network

FFD DAQ is similar to TOF DAQ, but it uses 2 VXS crates and 5 DREs per crate

Number of LTUs	1
Number of VXS Crates	2
Number of TTUs	2
Number of DREs	10
Number of data readout links	12
Data readout throughput (per link)	1 Gb
Number of White Rabbit links	3

Table 2: FFD DAQ Specification

4.3 ADC Based Readout

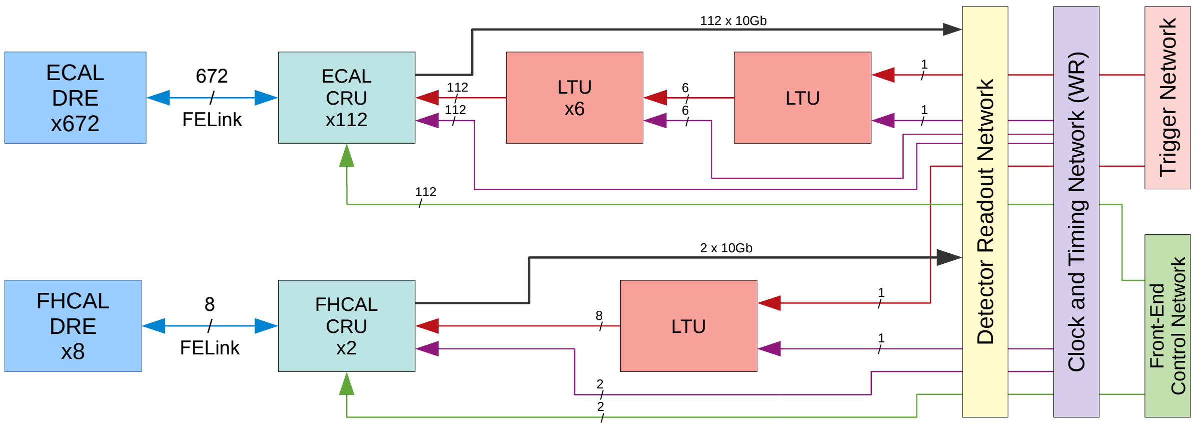


Figure 5: ADC Based Readout

As we assume, that LTU has 1 to 24 fanout, we have to cascade LTUs for ECAL. There are 6 second level LTUs in such model. FHCal is much more compact, so it has one LTU. 6 Data Readout Electronic boards are connected to Common Readout Unit (CRU) via FE-Link. One bidirectional FE-Link have data, trigger, FE control and clock recover information. It operates at 2.5 Gb/s speed. Each CRU unit have WR timing information. DRE boards recover WR clock from FE-Link, so they have constant time shift to CRU board. CRU uses 10Gb Ethernet to transfer all 6 DRE collected data to First Level Processor.

4.4 CRU Concept

The conceptual design of Common Readout Unit is shown on Figure 6. Hardware implementation: CTF-6 VXS board.

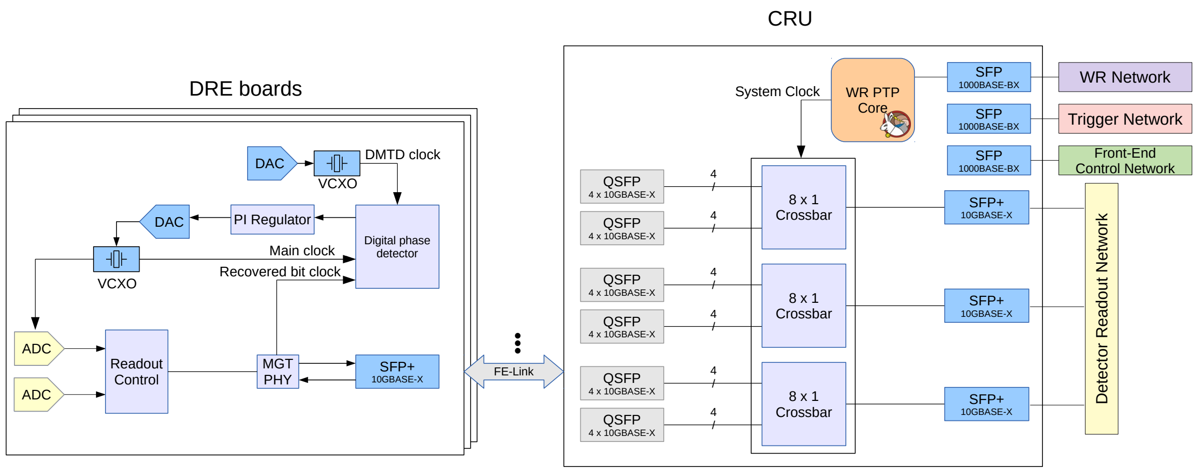


Figure 6: CRU – Common Readout Unit, Concept

4.5 Stage 2 Detectors DAQ

4.5.1 CPC Tracker

4.5.2 ITS - Inner Tracking System

4.5.3 Straw Tracker

5 Hardware Architecture

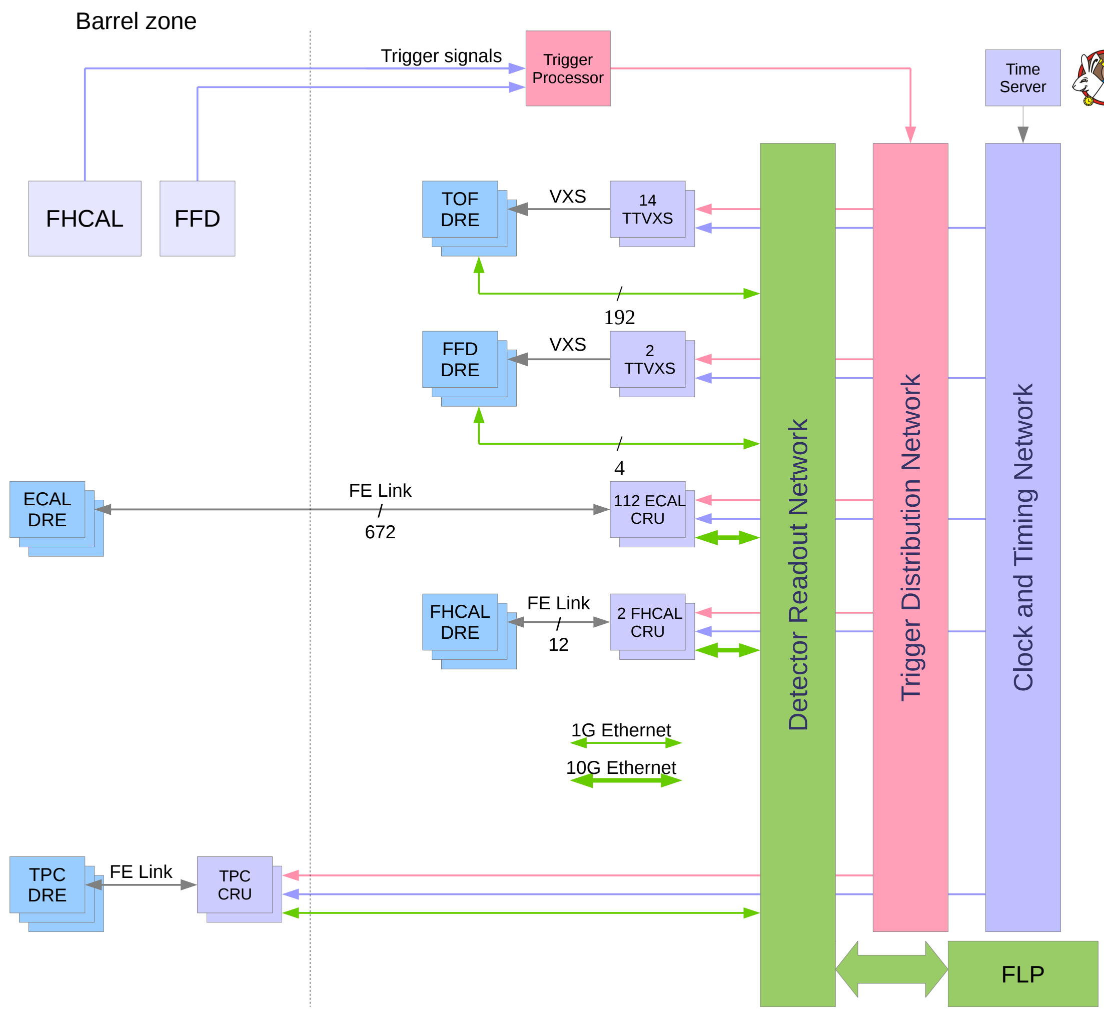


Figure 7: MPD DAQ Trigger, Timing and Data Links

The overall architecture of the DAQ (as shown in Figure 1) was designed from the start as a data driven fully parallelized push architecture. At each stage the content of the data stream drives autonomously its routing through the processing chain from the source to the destination. The throttling is implemented by a back pressure from the destination to the source.

5.1 Detector Readout Electronics

Detector Readout Electronics (DRE) boards record detector signals. There are three main types of DRE boards grouped by function: Timestamping Time to Digital Converters (TDC), Waveform Digitizers or Amplitude to Digital Converters (ADC) and Discrete Signal Counters (Scalers).

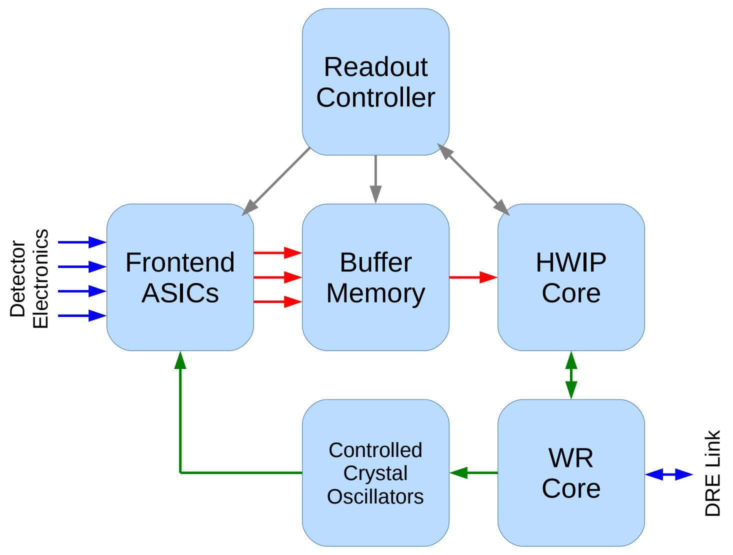


Figure 8: Detector Readout Electronics Board Structure

5.2 TDC Based Detector Readout Electronics

5.2.1 TTVXS

TTVXS is a Time and Trigger Unit (TTU) for VXS crate. It distributes trigger information and system synchronous clock to installed payload modules. Also TTVXS module collects busy information from all payload modules. Connection topology is star. TTVXS and each payload module connects with 1 serial bidirectional link with speed up to 2.5 Gb/s, 1 bidirectional LVDS link, 1 LVDS link for transmitting and 1 LVDS link for receiving. All LVDS links speed is up to 125 Mb/s.

TTVXS has SMBus link to all payload modules for monitoring its status. TTVXS can transmit this information via Ethernet 10/100M or CAN or RS485 interfaces.

TTVXS has 4 SFP+ connections for receiving synchronization and receiving trigger information from corresponding networks and send event data to data readout network. Connections speed can be up to 10 Gb/s. This module has 6 programmable input-output LVTTL coaxial connectors. Functions of this connection is not fully defined yet and it is subject to discus.

It is possible to install FMC card into TTVXS to expand its functions in future if its will be necessary.

Connection type	**Quantity**	**Description**
VXS	18	VXS links for TTC on VXS
Coaxial	8	6 Programmable LVTTL I/O, 1 Clock monitor, 1 Clock Reference
SFP	4	Trigger, Synchronization, Data Transfer
RJ45 Transformer Coupled	1	MCU Slow Control via 10/100Base-T Ethernet
RJ45	1	Monitoring via CAN or RS485 (TBD)
FMC	1	Multipurpose mezzanine extention

Table 3: TTVXS Interfaces

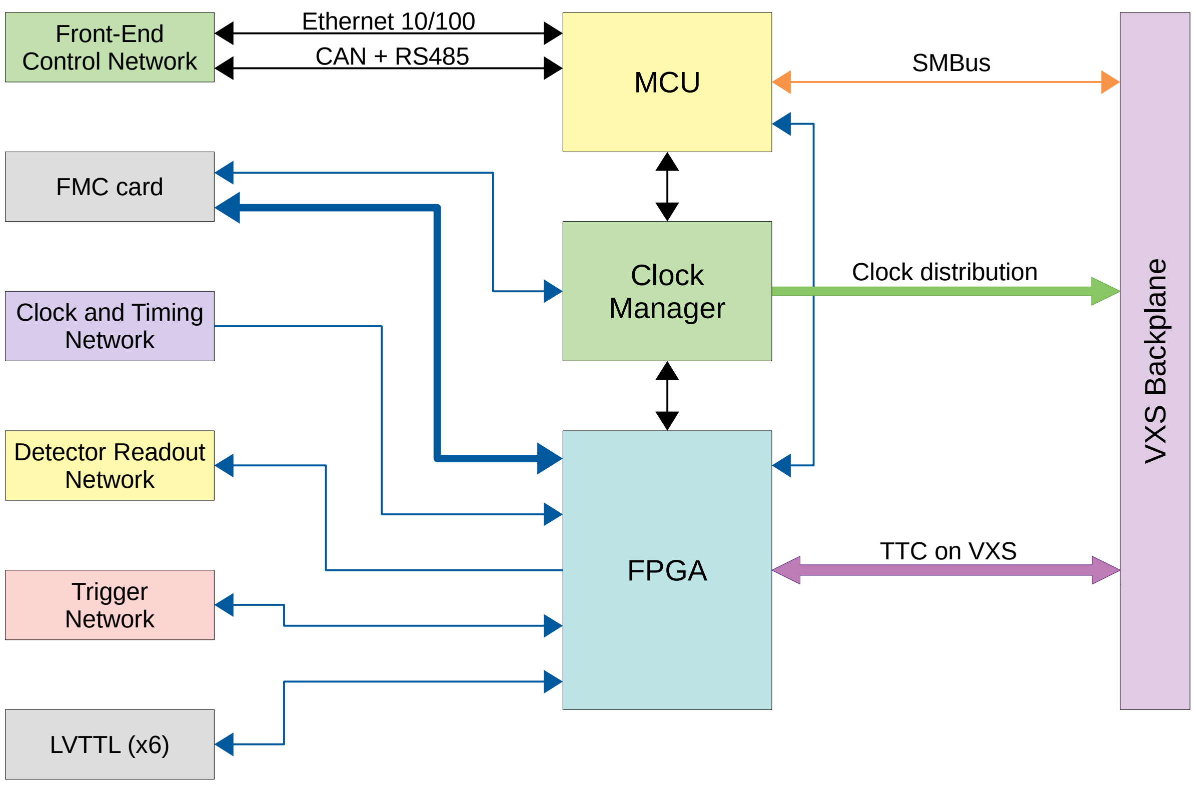


Figure 9: TTVXS Functional Diagram

5.2.2 TDC72VHL

TDC DRE board performs time-stamping of discrete signals (hits) with typical accuracy of 20 ps. It is based on HPTDC [2] chip. Hit timestamps are kept for 104 μs in ring type memory. The total trigger latency should not exceed this value.

Number of channels	72
Input signal	LVDS
Input impedance	100 Ohm
Input differential voltage	25mV min
Input connector	CXP Interconnect System
Time resolution with INL correction	20ps
Data transfer interface	SFP, 1000Base-X
Synchronization interface	TTC over VXS

Table 4: TDC72VHL Specification

5.3 ADC Based Detector Readout Electronics

ADC DRE board is a waveform digitizer. It quantize analogue input signal and samples it at fixed time intervals. Zero suppression logic is based on baseline estimation and threshold value. Signal shaping is performed in digital form with FIR filters. It allows to reduce the number of waveform points required for digital signal representation with minimum loss of accuracy. The ring type memory allows the read back of last 30 μs of waveforms. It sets the limit on trigger latency to this value.

5.4 Raw Data Streams

Raw data are pushed down as soon as available. All the participants of the data acquisition process from the DRE to the PDS, send the data through established connections to the next processing stage. The flow control is provided by TCP/IP (Transport Control Protocol) in event building network. DRE move data to LDC with the help of hardware implemented stream control protocol running over UDP/IP. The HWIP [3] hardware core is responsible for network interface in readout electronic boards.

5.5 MStream protocol

A custom design protocol Mstream is used for data transfers between detector's readout electonics and First Level Processor (FLP). It's main goal is continuous data stream fragmentation to defined size fragments for correct transportation. Random arrival fragments are combined in correct sequence to form initial data stream (events) on arrival side (FLP module). An ethernet UDP protocol is used as transport layer. UDP provides no guarantees for message delivery, so additional acknowledge was added for data quality check. The corresponding speed is about 65 MB/s for 1G ethernet and 270 MB/s for 10G ethernet. These numbers were obtained from GEM, STS, ECAL, ZDC and MWPC detectors during BMN runs in 2017-2018 years. A hardware HWIP Core is responsible for data movement from readout electronics to FLP. It has 16(8) data memory segments ring buffer. Each memory segment is at least 1.5 KB(8KB). It corresponds to maximum UDP frame payload size for standart Ethernet (or jumbo frame extention). Each sent fragment has unique constantly increasing sequence number, which is used in returning acknowledge frame from recieving side. If such acknowledge isn't recieved in defined time interval, corresponding data fragment retransmits. A flow control mechanism is used for data transfers. The main idea is to pause transfers if some fragment doesn't recive acknowledge during significant time. In other words, the difference between maximum and minimum frame numbers should not exeed double buffer counts. This allows to avoid big memory usage in FLP software.

5.6 Remote Firmware Maintenance

5.6.1 Multiboot overview

The possibility of easy firmware update is crucial for modern data acquisition electronics. The FPGA multiboot feature enables switching between images on-the-fly for remote updates. When an error is detected during the multiboot configuration process, the FPGA can trigger a Fallback that ensures that a known good design can be loaded into the device. The solution involves a flash memory that has a reserved areas to store these components:

- Fallback, or "golden bitstream"
- MultiBoot, or "update bitstream"

A remote programming method is implemented and is used to program new or enhanced bitstreams into the update bitstream area of flash memory.

5.6.2 Initial system setup

The multiboot image is first loaded at power up from an upper address space. If this image fails configuration (CRC error or watchdog timer), the device automatically triggers a fallback to the golden image stored at address 0. This enables systems to upgrade their own bit files and then boot from power up to the latest image. Fallback logic ensures the system recovers from any failure to load the multiboot image and loads the golden image. The golden image can then fix any errors in the flash and trigger a configuration from the multiboot image again.

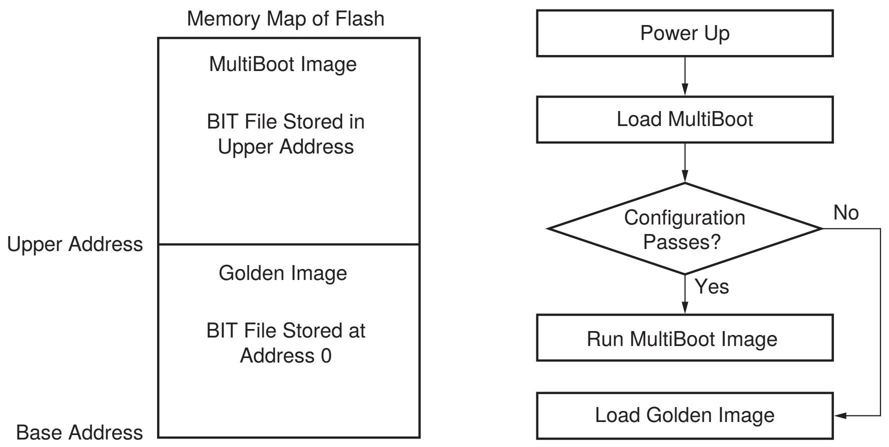


Figure 10: Initial MultiBoot Image Flow Diagram

5.6.3 Image update procedure

When a new image is compiled for update, it can be sent to device through Ethernet. The custom M-Link protocol (UDP with acknowledge) is used for this purpose. The FPGA then writes this image to flash memory using SPI (QSPI) protocol. It is possible to update image during normal operation. Power-up reset is the only requirement after update completion.

6 Timing System

White Rabbit [4] provides sub-nanosecond accuracy and picoseconds precision of synchronization for large distributed systems. It also allows for deterministic and reliable data delivery.

DRE boards digitize detector signals using common notion of time and frequency provided by the White Rabbit (WR) network. The time reference is provided by GPS/GLONASS receiver and backup precision frequency reference (Cesium or Rubidium clock).

6.1 Clock Precision

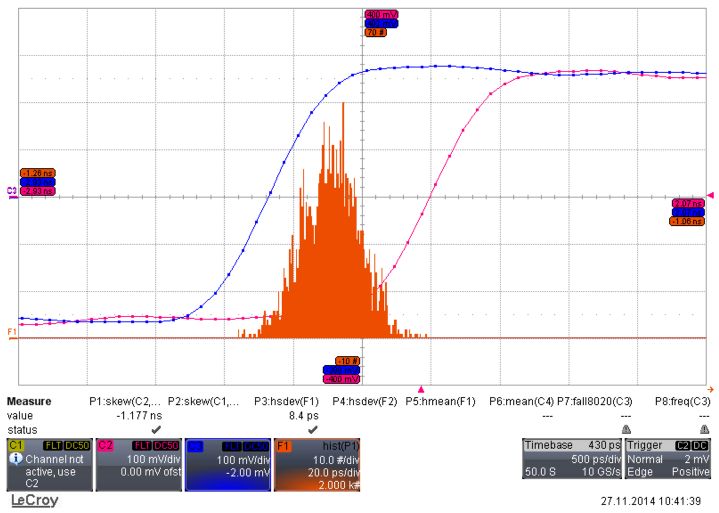
Digitizer boards require precise reference clock for high precision measurements. Timestamping TDCs used in FFD and TOF detector electronics have time resolution of $17psRMS$. The clocks in all DRE boards should be synchronized with $10ps$ precision for the measurement time.

DRE boards include White Rabbit Node Core [5] and tunable crystal oscillators that allows synchronization of all readout boards with sub-nanosecond accuracy and precision of $10ps$. The measured precision of synchronization between WR switch and WR node under test was $8.4psRMS$ shown on Figure 11.

6.2 Clock Synthesis

WR node core provides local clock and timestamp at 125 MHz. The timestamp is specified as TAI (International Atomic Time). It is an absolute number of seconds and nanoseconds since 01.01.1970.

Different types of readout board have specific requirements of reference clock frequencies. HPTDC ASICs runs at 41.667 MHz clock. Waveform digitizers runs at 62.5 MHz clock. Both frequencies are direct derivatives of 125 MHz WR clock taken from frequency dividers. The dividers are synchronized by 1 PPS (Pulse per Second) signal.



```
Measure   P1:skew(C2,...  P2:skew(C1,...  P3:hsdev(F1)  P4:hsdev(F2)  P5:hmean(F1)  P6:mean(C4)  P7:fall8020(C3)  P8:freq(C3)
value          -1.177 ns                        8.4 ps                                       ---           ---          ---
status                ✓                             ✓                                                      ⚠            ⚠
```

C1 FLT DC50 Channel not active, use C2
C2 FLT DC50 100 mV/div 0.00 mV ofst
C3 FLT DC50 100 mV/div -2.00 mV
F1 hist(P1) 10.0 #/div 20.0 ps/div 2.000 k#
Timebase 430 ps 500 ps/div 50.0 S 10 GS/s
Trigger C2 DC Normal 2 mV Edge Positive
LeCroy
27.11.2014 10:41:39

Figure 11: WR Node Time Synchronization Precision

7 Trigger System

7.1 Overview

The MPD sub-detectors FFD and TOF provide a set of L0 signals that are combined by trigger decision logic to form L1 trigger signal Figure 12 The latency of L1 trigger signal is constant.

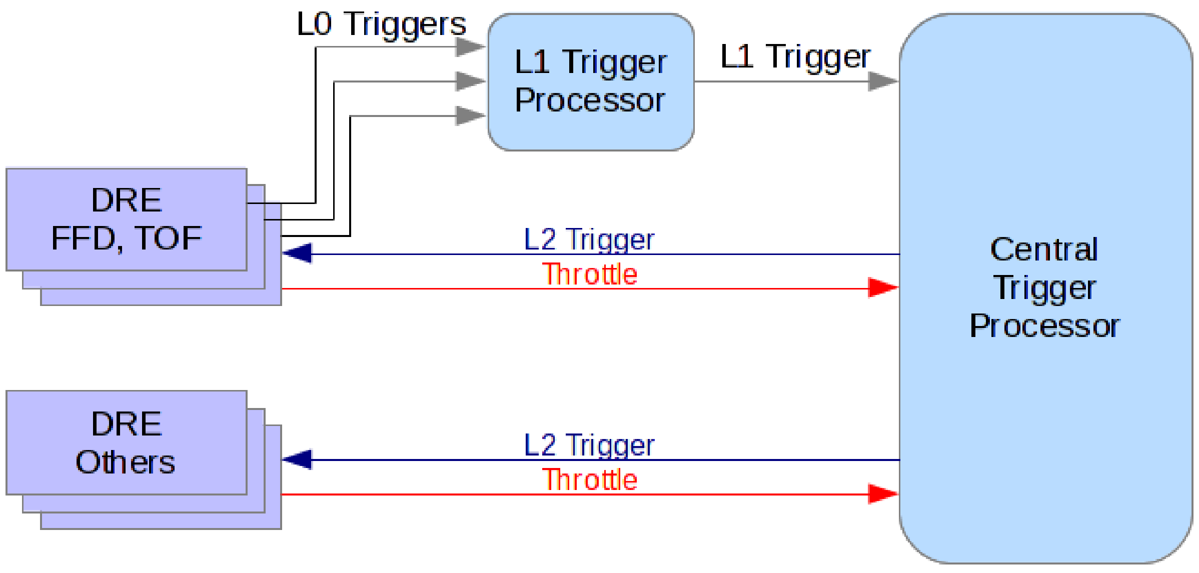


Figure 12: Trigger Levels

7.2 Central Trigger Processor

L1 trigger signal acts as a source for L2 trigger. Central Trigger Processor performs the time-stamping of incoming L1 trigger signal, encodes it in a sequence of high priority Ethernet frames and sends to Trigger Distribution Network. Throttle signals are periodically sent by Local Trigger Units to CTP. CTP drops L1 triggers when the whole system is at a limit. It occurs when any of the DRE memory buffer is nearly full. This provides consistent data taking and protects from unexpected raw data loss in DRE ring memory or interconnecting networks. Thus the raw data from DRE is moved to Event Builders by L2 trigger.

7.3 Local Trigger Unit

The Local Trigger Unit (see **??**) communicates with readout electronics (DRE) through dedicated high-speed synchronous link. Taking into account the delay on endpoint serializers and deserializers, packet assembly and decode, the total delay from LTU to DRE or in opposite direction will be below $5\mu s$. This is sufficient for both L2 trigger and throttle signal transfer.

7.4 Trigger Distribution

L0 and L1 triggers are distributed through coaxial cables. The L2 Trigger distribution system is based on real-time data transfer over dedicated serial links. The series of L2 trigger signals within fixed time period is time-stamped, encoded and sent over trigger link as sequence of frames. Forward Error Correction (FEC) technique provides reliable transport. On the receive side, the frames are decoded and trigger information extracted. The L2 trigger information is used to select required data specified by the time stamps from the ring buffers in detector readout electronics. See [6].

7.5 Trigger Control Loop

The principle of the MPD DAQ read-out structure relies on the Trigger System distributing the L2 trigger signals with guaranteed maximum latency to the read-out and front-ends and to receive the busy signal to throttle the trigger distribution if necessary. Trigger and busy signals are time tagged and constant latency link is not required. The maximum allowable round-trip time of L2 trigger processing, distribution and busy signal return is limited by the front-end electronics memory size and is $30\mu s$.

8 First Level Processor

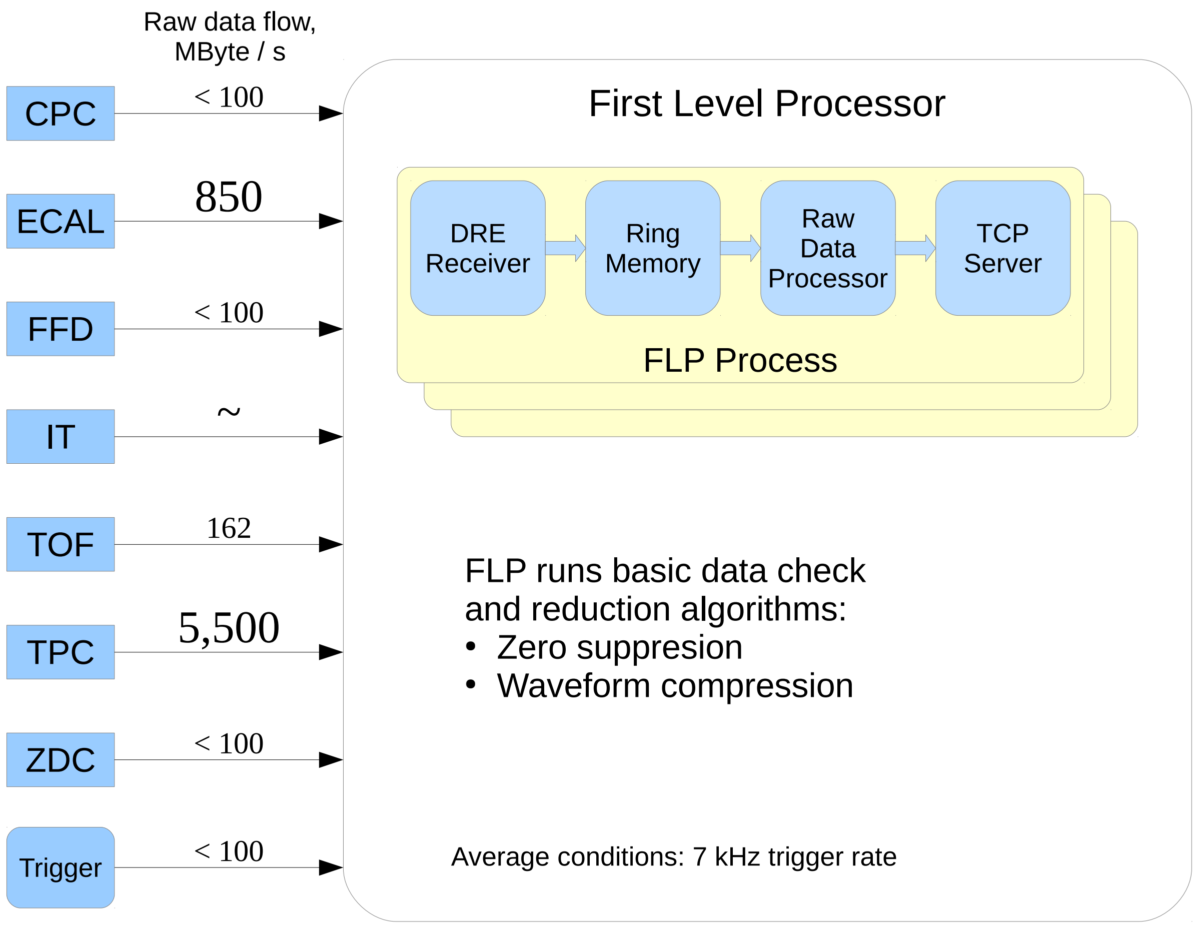
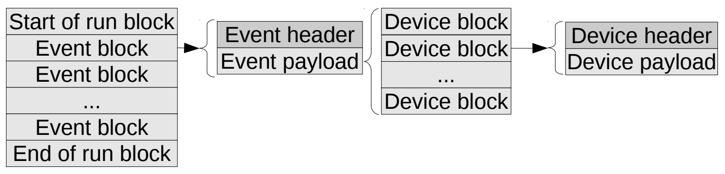


Figure 13: FLP Structure

8.1 Raw Data Format

Detector readout electronics send data to the First Level Processors in MPD-Raw-Data format. There are two main levels in it. Event and device blocks. This format is using TLV (type-



Event header	Bits
Synchro word	31:0
Payload length	31:0
Event number	31:0

Device header	Bits
Serial Number	31:0
Device ID	31:24
Paylload length	23:0

Figure 14: MPD Raw Data format

length-value) architecture. Each event blocks started with following header:

Sync word is using first of all for synchronization of data flow. If there will be some losses we can find beginning of *MPD data block* by this synchro-word. Also one can define different

32-bit word	Description
0	Sync word
1	Event payload length, bytes
2	Event number

Table 5: MPD Event Header

types of events, and furthermore version control can be implemented. At right moment there are four *sync word*:

- 0x2A502A50: events with physics data (regular data events)
- 0x4A624A62: events with calibration data
- 0x72617453: start of run block
- 0x706F7453: end of run block

8.1.1 Event block

Event payload is constructed with sequence of device blocks, e.i. each *Event payload* started with Device block, which started with *Device header*:

32-bit word	Byte offset	Description
0	31:0	Sync word
1	31:24	Event payload length, bytes
	23:0	Event number

Table 6: MPD Device Header

Format of device's payload defined by device type (Device ID). It is known in First-Level-Processing stage, but in there is no need in it at Event-Building stage.

8.1.2 Start/stop of run block

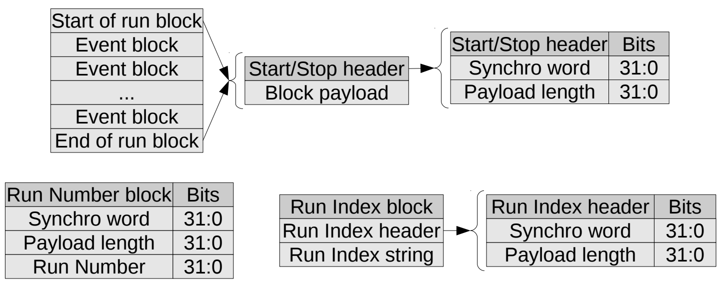


Figure 15: Start/stop of run block structure

Both *Start of run block* and *Stop of run block* has same structure, they differ only in *Sync word*. Start/stop payload consists of two TLV-blocks:

- 0x236E7552: Run number block
- 0x78646E49: Run index block

Length of *Run number block* is 4 bytes. 32-bit unsigned integer (*Run number*) is more than enough to number runs.

Run index payload is *Run index* string in the Latin-1 encoding. If the length of the *Run index* is not a multiple of 4, then string appends with NUL.

9 Event Builder

9.1 Introduction

Data combiner system task is to receive data flow from First Level Processor (**FLP**) fabric, create events and store then in Transient Storage System on Online Cluster (Figure 1)

Event Builder components communicate with each other by TCP/IP. The protocol is defined in MPD-Raw-Data format section (see 8.1 Raw Data Format).

The base unit of DAQ software is Event-builder (EvB) program that receives data-flow from TCP servers, assembles events using event numbers and transmits resulted event block to configured output. In terms of MPD-Raw-Data format its concept is shown in Figure 16.

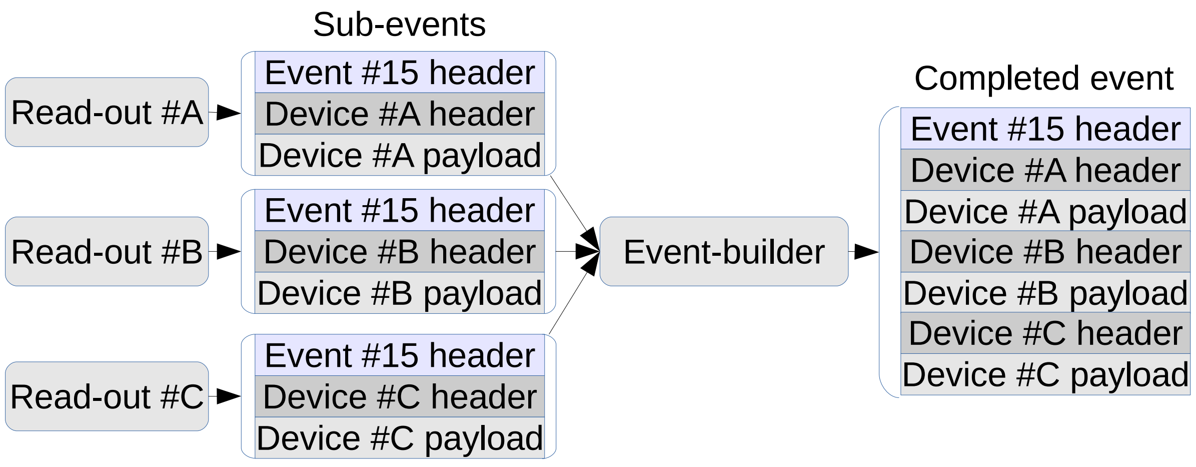


Figure 16: Concept of Event-building system

9.2 Event Builder Architecture

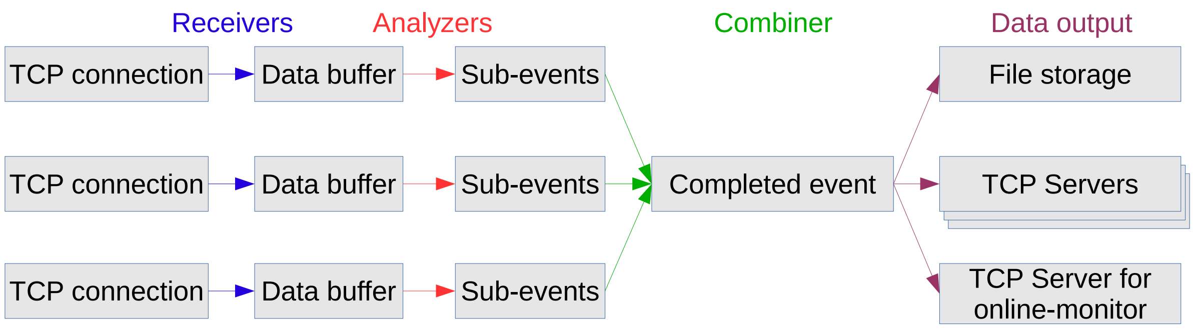


Figure 17: Single Event-builder architecture

There are four main parts of event-builder:

- Receivers from read-out programs.
- Analyzers of received data.
- Combiner of completed event package.

- Output of completed event package.

There are separate TCP receivers, ring-buffers for raw data, data stream analyzers and sub-events meta-info buffers per each TCP-connection. And there are separate ring-buffers for completed events and writers (file-writer or TCP Server) per each output option. Each part: TCP receivers, data analyzers, combiner and all outputs work in separated threads and works asynchronously.

9.2.1 Receivers

Receiver's task is to establish TCP-connection, receive data-flow from it and write data into its separate data ring-buffer. Receiver doesn't perform any data check. Receiver will work, provided if there are some free space in data ring-buffer and pending data in TCP-connection.

On program start it connects to TCP-server of read-out program. And start receiving data while there is sufficient space. In case of data ring-buffer is full it will stop receiving data from TCP-connection (thus performed back-pressure of data-flow) and wait until analyzer free sufficient space.

9.2.2 Analyzers

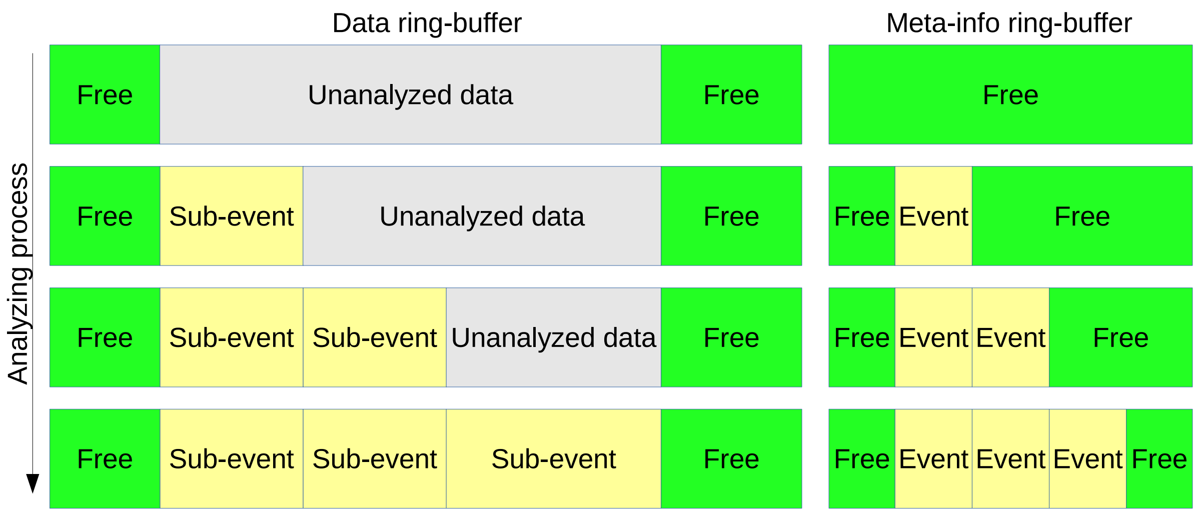


Figure 18: Analyzer's work in EvB

Analyzer's task is to look after received data and distinguish sub-events data blocks. Start of sub-event block is defined by synchro word in it's header. Ending is calculated by length in block header. Normally, data flow starts with sub-event block header and there will be another header after block ending. If analyzer got unknown syncro-word it means that we have data flow corruption, and after that it starts byte-by-byte shifting from last correct event ending until it reached MPD-event header with known syncro-word.

Once sub-event block was detected, analyzer write meta-info into meta-info ring-buffer based on this sub-event.

Meta-info structure:

- Event type (End-of-burst or payload data)

- Event Number
- Pointers to start and end of the data block
- Creation timestamp
- Clear flag

In background analyzer clears freed space in ring-buffers: if combiner finished processing of some sub-event it would release meta-info structure; once analyzer detected released meta-info free data block in data ring-buffer and remove this meta-info from meta-info ring-buffer. Analyzer doesn't copy raw data, just read MPD Event Header.

9.2.3 Combiner

There are several rules for combiner:

1. Calibration events passes without building. It's data directly transmitted to the output.
2. Data with event number lesser than expected event number are dropped (it was late).
3. Event is completed if all analyzers provided data with expected event number.
4. Combiner can't wait longer than specified timeout from earliest creation timestamp in list of received meta-info.

Normal work of combiner will be following: wait for sub-event from any analyzer → wait for missing sub-events within timeout → build completed event based on received sub-events → send it to the output → repeat again.

This work can be broken only if some sub-event are delayed relatively to the fastest one. In this case event will be transmitted to the output uncompleted. And combiner start waiting sub-events with next event number. All delayed sub-events will be dropped by rule #1.

9.2.4 Output

The event builder process has tree output options:

- File storage
- List of TCP servers
- TCP server for online-monitor

EvB can be configured to transmit all events to the File storage or to distribute events among list of TCP servers. These options should transmits all combined events. If there will be lack of free space in output, combiner will paused until sufficient space will be freed.

The third option is optional. EvB will transmit event to it only if there is established connection and there is enough free space in its buffer, unlike the first two options. Each event can be transmitted either to *File storage* or to the TCP server.

9.2.5 File Storage

In File storage mode, completed event should be written in specified file. By default EvB write separate file for each subsequent run.

In case of big data flow (see 9.3 Multi-staging on the Global Data Combiner), all data flow will be split into several files.

Also it is possible to setup slice options (see 10 Data Slicing). If slicing will be enabled then every EvB will wrote separate file for each slice.

9.2.6 Blocking TCP Stream

In Blocking TCP stream mode, event builder sends completed event to output TCP-server. If there is no free space in buffer of TCP-server the combiner thread will be paused until enough free space will be available.

Although there can be not single but list of output TCP-servers. Completed event will be transmitted through TCP-server which index will match event number of completed event.

With this option we can complex event-building system (see 9.3 Multi-staging on the Global Data Combiner).

9.2.7 TCP Online Monitor

If TCP online-monitor mode is enabled, completed event can be sent to monitor TCP-server if there available space. If monitor program is not too fast to proceed all data from TCP-server, combiner would skip sending completed event. That's why not all events might be transmitted through this channel.

9.3 Multi-staging on the Global Data Combiner

In final stage of experiment there will be hundreds of detectors and one event-builder couldn't process data from all read-out programs. In this case we can split event-building process into two logical levels.

In first level we will combine all events from rather small groups of read-out programs. Event-builder *Group #A* will combine event data block from read-out programs 1-3 while *Group #B* from read-out programs 4-6.

Second level event-builders will combine event data with specific event number from all first level event-builders. If there are $\boldsymbol{K}$ event-builders in second level, Event-builder $\#\boldsymbol{i}$ will combine event data with event number $\boldsymbol{EvN}$: $\boldsymbol{EvN \bmod K = i}$.

Event-builder #0	Event numbers: 0, K, 2K, 3K, ...
Event-builder #1	Event numbers: 1, K+1, 2K+1, 3K+1 ...
...	
Event-builder #K-1	Event numbers: K-1, 2K-1, 3K-1, 4K-1, ...

This architecture allows us to combine events with small data from hundreds of detectors (by increasing number of event-builder from first level) otherwise context switch decrease EvB performance. And consume huge data flow (by increasing number of event-builder from second

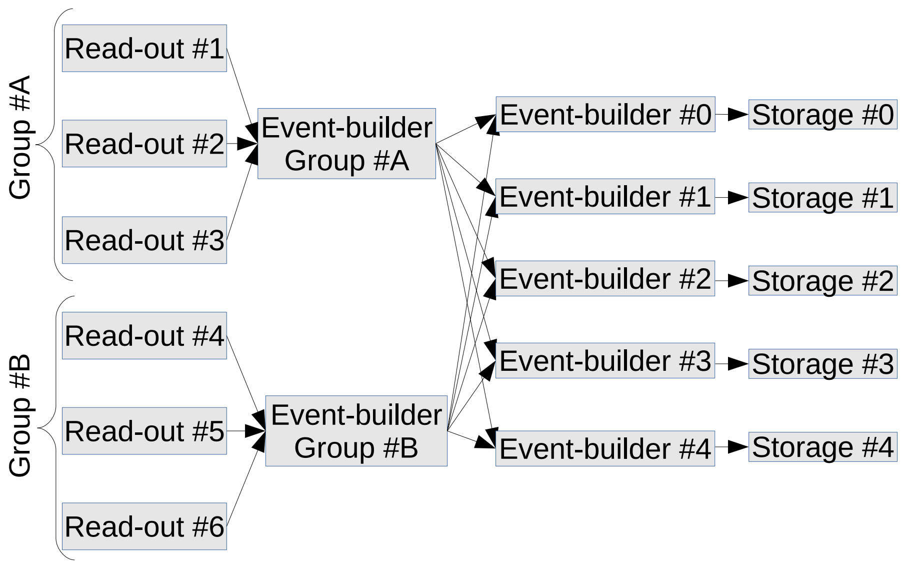


Figure 19: Global architecture of EvB system

level). In result each event will contain data from all read-out programs and will be stored in one of the storages (all events are equivalent).

9.4 Data Integrity Check

EvB knows only MPD-Raw-Data format and can't check device's payload that why they can check that all configured connections to read-out programs send their part of event data. On each trigger all read-out programs should send event data (in case it has no data to send, event block with empty payload is sending).

9.5 Event Builder and DAQ Runs

Each data-taking period will be partitioned in sequences of events taken under constant running conditions. These periods are called 'runs'. The event-builder process is instructed to perform special functions at the beginning and at the end of each run.

10 Data Slicing

Data slicing is performed in continuous run. It allows to split output data-file(s) by time or by the number of events.

Data slicing configurator communicate with EvB and allows to configure it to split output data-file(s) by time or by the number of events.

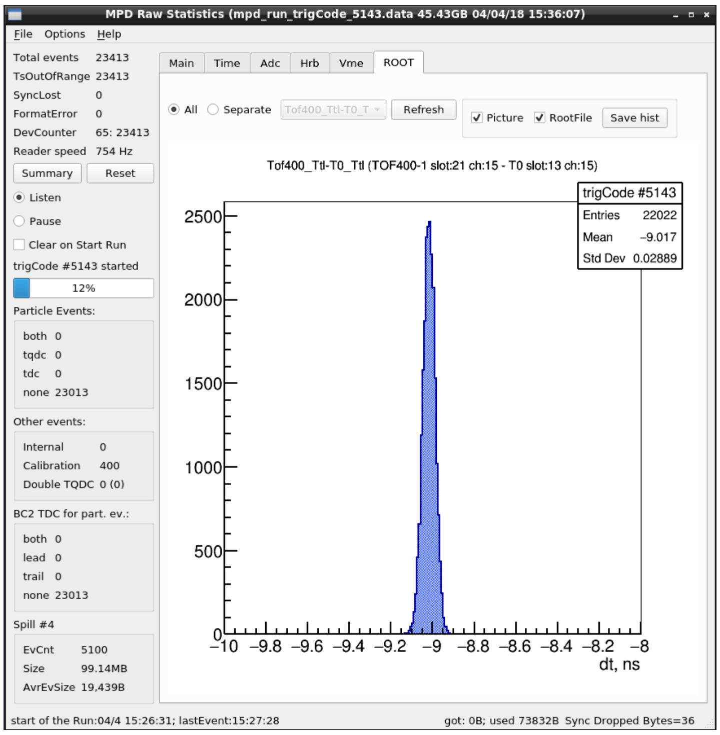
11 Online data check

Online data check is performed with *MPD raw statistic* program. It can use either data file or monitor port of EvB as source of raw data. It knows internal formats that is used in DAQ system and can perform complex check.

Main checks are following:

- all DAQ format is valid and can be decoded
- every event has same count of devices
- WR-time in all devices has suitable distribution
- every event has T0 time mark
- collect some metrics and transmit it to the storage

MPD raw statistic can histogram data values, such as difference of time between some channels.



```
MPD Raw Statistics (mpd_run_trigCode_5143.data 45.43GB 04/04/18 15:36:07)

Total events    23413
TsOutOfRange    23413
SyncLost        0
FormatError     0
DevCounter      65: 23413
Reader speed    754 Hz
(●) Listen
( ) Pause
[ ] Clear on Start Run
trigCode #5143 started
12%
Particle Events:
  both  0
  tqdc  0
  tdc   0
  none  23013
Other events:
  Internal      0
  Calibration   400
  Double TQDC   0 (0)
BC2 TDC for part. ev.:
  both  0
  lead  0
  trail 0
  none  23013
Spill #4
  EvCnt       5100
  Size        99.14MB
  AvrEvSize   19,439B

(●) All  ( ) Separate  Tof400_Ttl-T0_T   [✓] Picture  [✓] RootFile

Tof400_Ttl-T0_Ttl (TOF400-1 slot:21 ch:15 - T0 slot:13 ch:15)
trigCode #5143
Entries   22022
Mean      -9.017
Std Dev   0.02889

start of the Run:04/4 15:26:31; lastEvent:15:27:28      got: 0B; used 73832B  Sync Dropped Bytes=36
```

Figure 20: MPD Raw Statistic screenshot

12 Data rootifier

Data rootifier receives information about completed files from EvB performed in run. Its task is to save raw data into ROOT-file with help of MPD-ROOT framework in Transient Data Storage.

13 Data mover

Data mover receives information about completed ROOT-file as result of *rootifier* work (see 12 Data rootifier). It should push them to *Permanent Data Storage*. *Data mover* should receive MD5 hashes as acknowledge that files received successfully and after that delete ROOT-file from Transient Data Storage.

14 Run Control

14.1 Run Control Overview

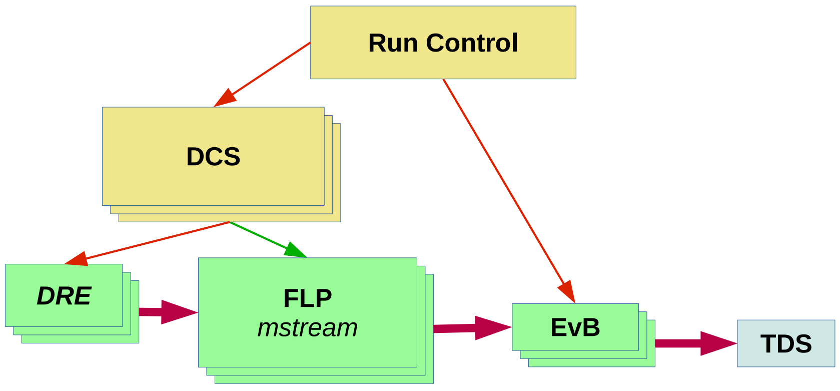


Figure 21: DAQ software

Run Control is a program that controls all elements of DAQ. Elements are of two types Device Control System (DCS) and Event Builder(EvB). Run Control is a GUI-program that can be controlled with GUI-interface or with updates of Configuration Database. First type of control is preferred when system works as indivisible whole, second one used when there is partitioning of the system.

14.2 Device Control System

Device Control System is a program that can interconnect with specific type of DRE (ex. ADC64 Control System, TQDC Control System, etc.) and setting it up. Also DCS can start and look after FLP programs for each device of a system.

14.3 Run Control States

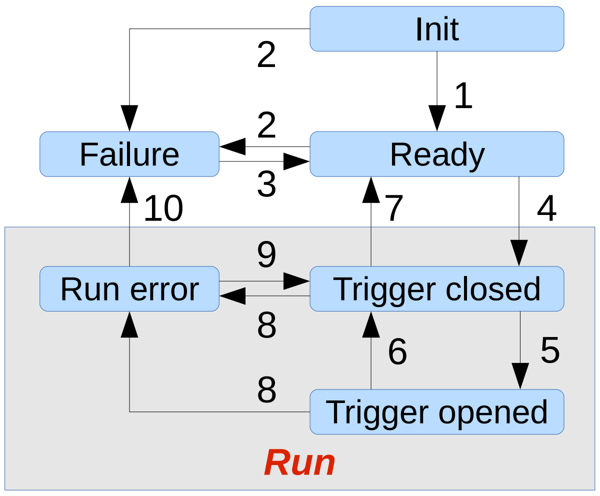


Figure 22: Run Control states

Failure and ***Ready*** are states in which we can change Run configuration (in these states we out of Run).

While ***Trigger closed***, ***Trigger closed*** and ***Run error*** are states in which Run is executed, and Run configuration is fixed, and we can manipulate just with Trigger state.

Run Control states are:

1. ***Init*** is initial state. The Run Control connects to Configuration database and loads current configuration (see 15.1 Configuration Database).
2. ***Ready***. In this state Run Control has established connections with all clients and they are ready to start a Run.
3. ***Trigger closed***. All clients have made all preparations and ready to process event data. But distribution of trigger is blocked.
4. ***Trigger opened***. Triggers are distributed, DRE boards produce data, events are combined and transmitted to TDS. Entire DAQ system works fine.
5. ***Run error***. An error occurred in run preparation/execution. System trying to fix this error.
6. ***Failure***. System can't start/continue a Run execution with current configuration.

Run Control can change its state with following actions:

1 Run Control has established connection to Configuration Database and loaded configuration is correct.

2 Run Control has no valid configuration, i.e either one can't receive current configuration or current configuration is invalid.

3 Run Control obtain valid configuration, i.e. either RC restore connection to Configuration Database and load configuration, or all elements of DAQ system became available and ready for work.

4 Run control sends to all clients current configuration and they response that they applied configuration and ready to process event data flow.

5 Opening trigger. DAQ works in accordance with Run config and start proceeding events.

6 Closing trigger. RC stop distributing triggers. All system is ok.

7 Closing Run. RC is reporting about end of Run. All clients wait when all data will flushed and all files will be closed. Only thereafter clients report to RC that they successfully closed Run.

8 Some error occurred while Run is executed.

9 DAQ system fix occurred error and ready to continue Run execution.

10 DAQ system failed to fix occurred error, trigger closed and all clients quits from Run execution.

For correct run closing (action #4) we use following mechanism: trigger distribution is closed, and RC can find out latest event number, then it inform EvBs from second level that run is closing and latest event number is the one it found. As soon as EvB combined and flushed its last event it report back that this EvB has finished its Run work. When RC received reports from all EvB from second level, one declared that Run is closed, all data was transmitted to TDS and DAQ system is ready to change its configuration or to start new Run.

15 Computing Services

15.1 Configuration Database

Many parameters control the DAQ operation. The configured number of DRE modules, event builders and other components information is stored in Configuration Database. Detector permanent parameters such as number of channels and mapping of detector to front-end and to readout channels are stored in database.

15.2 Message Logging

Error, warning and information messages from all of the software subsystems and infrastructure elements are transferred to the central logging service. The purpose of logging service is to allow real time analysis of past and current operating conditions, prediction of incorrect operation and failures.

The usage scenarios of log message analysis usually includes search for small but valuable information by excluding known and unimportant messages with search criteria iteratively. The search terms are not known in advance. This is like searching the needle in a haystack.

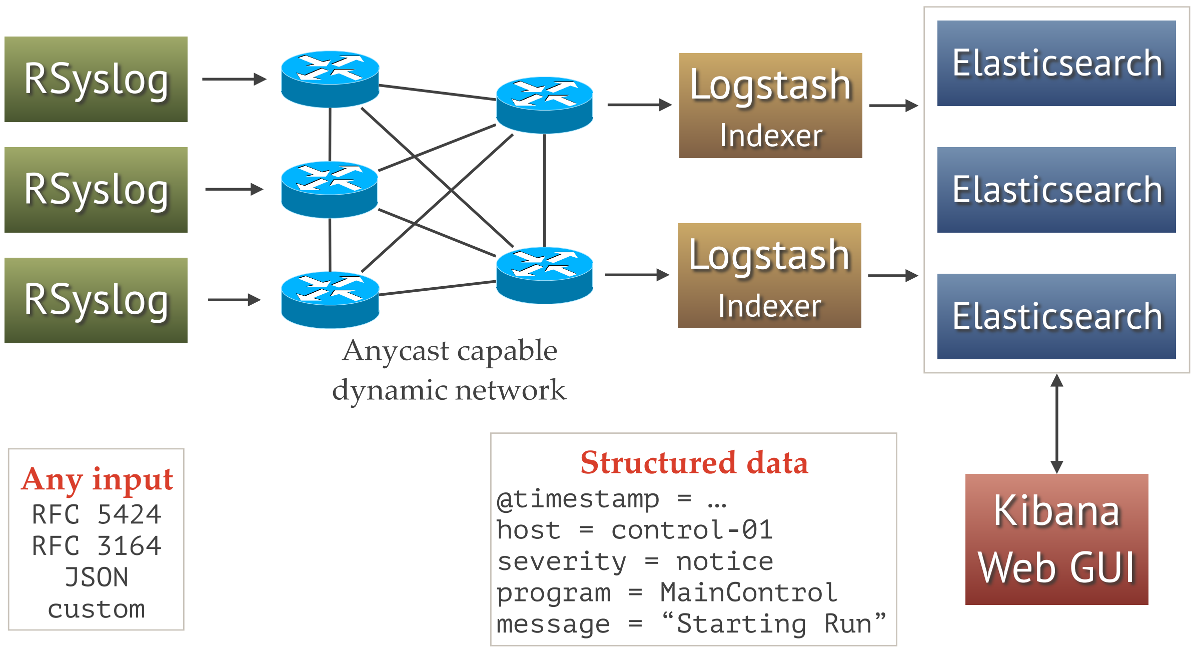


Figure 23: RSyslog+OSPF+ELK High Availability Logging

The logging system is comprised of the following parts:

1. RSyslog - Message receivers that register syslog messages by rsyslogd daemon processes running on every host.
2. Dynamic network routing protocols and routers – Highly Availabile message distribution
3. Logstash – Message parsing and forwarding to Elasticsearch storage cluster
4. Elasticsearch cluster – indexer and replicated storage

5. Kibana – Web interface for querying and analysis

Any type os unstructured data is acceptable, however for deep analysis, correlation of events and for quantitative reports it is necessary to parse incoming information into predefined strucured format. The simplest approach is using key-value representation of data, where key is predefined tag and value is text, number or other formatted information (date, time, Geo location, IP address, etc.)

It is common for the task of logging to use BSD Syslog Protocol. After considering the alternatives we have chosen RSyslog system as well supported, documented and flexible. It is and open source program that implements processing of RFC-3164 BSD Syslog Protocol, RFC-5424 and many other standards.

The messages from DAQ programs as well as other system messages are passed with syslog system call to rsyslogd daemon that should be running on every DAQ host. Many other systems, including newtorking equipment, VME, mTCA and server chassis, UPS (Uninterruptible Power Supplies) are capable of using syslog standard for logging system messages.

Upon reception of system message, the rsyslog daemon store it in internal buffer and on persistent storage (file). rsyslog daemon has multiple configurable output options and DAQ system uses pass-through mode with RFC-5424 output standard and TCP connection to destination logging system.

The destination of syslog messages is single IP address. The drawback of using single physical destination host vulnerability to the failures of destination system. Since DAQ services are required to be High Available, the IP address virtualization and cluster synchronization techniques could be used, for example HAProxy with Corosync. This is well supported and robust solution used widely for web services by cloud operators. The drawback is the requirement of cluster synchronization protocol that imposes restrictions on the network structure by requiring Layer-2 connectivity between hosts.

There is an alternative solution that is well scalable, an IP Anycast. Anycast is a network addressing and routing methodology in which a single destination address has multiple routing paths to two or more endpoint destinations. Routers will select the desired path on the basis of number of hops, distance, lowest cost, latency measurements or based on the least congested route. Anycast networks are widely used for CDN products to bring their content closer to the end user.

Since the DAQ network is required to be redundant and Highly Available, one can use it for syslog message routing also. This requires the support of dynamic routing protocols on receive side of Logstash nodes. In DAQ network the dynamic routing is performed with OSPF (Open Shortest Path First) protocol. Logstash nodes has Quagga software that implements network routing protocol in software. The OSPF routing daemons have established connections to the neighbour OSPF daemons in network equipment and announce the logging destination address to the network. The scripts track the availability of logging service and shuts down OSPF daemon in case of problems thus allowing service takeover by remaining Logstash nodes.

The task of Logstash nodes is to parse incoming message using predefined rules and to dispatch formatted message to Elasticsearch cluster. Each of the Logstash nodes has connection to multiple of Elasticsearch nodes for redundancy and load balancing.

The Elasticsearch cluster is comprised of high performance servers with solid state storage elements to allow indexing of incoming messages with peak rate of 100,000 per second. Although the average message rate is not very high, 1-10 per node, this number increases in case of failures and unexpected situations. The short peaks in message flow activity are also flattened by buffers in every stage of message processing, in RSyslog and in Logstash hosts.

The other performance requirement for Elasticsearch nodes come from data querying requirements. Search queries executed by user with Kibana Web GUI are run on all Elasticsearch cluster nodes in parallel. In order to complete search query in reasonable time it is important to these nodes to have enough processing power.

The prototype of Message Logging System has been designed, built and is operational with limited performance since Septemper 2015. It has been successfully used for troubleshooting and is used as a primary source of system status information in everyday operations.

year	2017	2018	2019
Data availability period, days	300	365	365
Average data flow, messages per second	115	180	300
Monthly index size, GB	200	300	500
Database size, GB	2900	4300	7300

Table 7: Logging System Parameters

15.3 Monitoring and Alarm System

The System Monitoring purpose is continuous collection of quantitative metrics and system status information, its analysis and generation of alarms using predefined trigger conditions.

There's many systems on the market as well as open source systems. We have used MRTG, Cacti, Nagios, OpenNMS and other monitoring systems and have found that the most suitable for the purpose of DAQ system monitoring is Zabbix at the moment of writing.

Zabbix is an enterprise-class open source distributed monitoring solution. It is a software that monitors many parameters of infrastructure and networking equipment, servers, software systems. It has a flexible user notification mechanism (alerting) based on configurable trigger conditions. It supports both push and poll data reporting mechanisms. The web interface provides table, graph and trend reports that are used for historical analysis.

Monitoring system is comprised of these parts:

1. zabbix agents – software that runs on monitored hosts and collects metrics
2. zabbix proxies – software that polls equipments and store metrics in intermediate database
3. zabbix server – central component that receives data from proxies, stores in database, reports statistics and alarms.
4. database – stores historical and configuration data
5. web interface – provides user access

The monitoring system is not required to be Highly Available, however the outages of central components (zabbix server and database) are mitigated by using proxies with intermediate databases.

The prototype of Monitoring and Alarm system has been designed and put into operation in 2012. Since that time it is used as a main monitoring tool on eveyday basis and during BM@N runs.

The parameters of the monitoring system prototype are summarized in Table 8. Upgrades in 2018 and 2019 years are planned to satisfy increasing number of monitored hosts and keep table and graph query time under reasonable limits. It is estimated that start of the MPD DAQ system operation will case the increase of the number of monitored metrics by the factor of 2.5

year	2017	2018	2019
Historical data availability period, days	30	365	365
Historical information per day, GB	12	32	81
Trend information volume, GB per year	400	530	1300
Query time for 100 historical metrics, seconds	20	3	3
Total number of metrics	150,000	200,000	500,000
Total number of trigger conditions	130,000	180,000	450,000
Average item query period, seconds	120	60	60
Information flow, items per second	1300	3500	8750
Database size, GB	760	12,300	30,800
Database storage	SATA, Ceph	NVMe RAID	NVMe RAID

Table 8: Monitoring and Alarm System Parameters

15.4 Performance Monitoring

Each program of DAQ system always calculate its own statistic. This statistic allows to control every part of DAQ system. We can detect what part of DAQ system has maximum performance load, where is overflowed buffer, what is event rate, what is event size, etc. We can draw charts of these metrics to visualize whole work of DAQ system.

15.5 Data Integrity Monitoring

There are two parts of monitoring. First of all EvB checks that all incoming channels has provided its sub-events on every trigger, otherwise error message will occurred. Furthermore online-monitor mode of EvB can sends completed event to Integrity Monitor program, which can check that all devices have provided correct payload according to it data format. Also it can perform some specific check, for example that timestamps from all devices correspond to one event, i.e. checking of events mixing.

15.6 Event Builder Testing System

EvB is rather complex system and to know its limitations we use Event Builder Testing System.

First element of this system is MPD Test Generator. This program should emulate output of FLP program, i.e. produce events with random generated data. Event data will be generated with pseudorandom generator. But program will construct event data not only with generated data, but also prepend md5 hash value of this data and seed number in order to be able to complete check data corruption at the end of EvB system. Event Generator use following options:

- Device id
- Device serial number
- List of work intervals:
  - Interval's start time
  - Interval's finish time
  - Number of events that should be produced in this interval
  - Interval in which data length should be.

MPD Event Generator exits when there is no pending work intervals.

Second part of this system is MPD Test Analyzer. This program will analyze a data from EvB system. It will know options of generators that was used to produce this data. And it should check that all data is valid, i.e. received data is the exactly one that was produced by MPD Test Generator. MPD Event Analyzer exits after proceeding of all planned events.

Third part of this system is MPD Test Manager. First of all this program should create a plan of upcoming test:

- How many MPD Test Generators will be used in test
- Configuration of each MPD Test Generators
- EvB system structure:

  - How many first level EvB will receive data from MPD Test Generators
  - How many second level EvB will receive data from first level and sends to output

- how many MPD Test Analyzers will proceed data from EvB system

And it should start all these programs before start time of the first interval. After that it waits till the end of the last interval and close all started EvB.

16 DAQ Networks

16.1 Detector Readout Network

Detector Readout Network connects the readout boards to data concentrators and receivers. The links originate in electronic boards located either inside MPD barrel and some terminates in MPD platform racks or in counting room racks. Data concentrators (CRUs) are realized with custom designed electronic modules with programmable logic and performs link aggregation function. For subdetectors that use standard Ethernet encapsulation for data transport the use of commercially available switches is possible assuming they implement all necessary functions.

Detector Readout Network main function is low delay, lossless and near real-time data transfer. Also it provides a transport for front-end electronics configuration, control and monitoring.

The data flow from DRE is realized by custom data streaming protocol: M-Stream. In terms of OSI model, M-Stream is application level protocol and it is encapsulated in UDP. It's main constraints are restriction of hardware implementation on FPGA and requirement of high throughput.

The detector readout network topology is designed using physical constraints and data throughput requirements. The number of required ports is defined by the number of detector readout electronics boards, by detector logical partitioning and data throughput. The number of required links to connect Detector Readout Electronics is summarized in Table 9.

Detector	10 GbE data links	1 GbE data links	1 Gb WR links	Data Link utilization, %
ECal/B	120	-	112	0.8
FFD	2	-	2	
FHCal	12		2	
TOF/B	-	192	12	0.9
TPC	24	-	24	15

Table 9: Subdetector network connections

Number of interconnects with low data throughput are aggregated by switches in tree like topology. The total throughput of aggregated links should not exceed the capability of upstream links to ensure loss-less operation of data transfer networks. Under normal conditions single link utilization is expected to be kept under 60 % for reliable, loss-less operation. This takes into account the UDP- based protocols used for communication between detector readout electronics and First Level Processor. The packet retransmissions should be avoided because it leads to delays and unnecessary throttling of trigger (dead time caused by link congestion). Moreover the packet retransmission itself increases link utilization factor.

The network topology design accounts for upgrade possibility. A factor of 2 is reserved for bandwidth in aggregated interconnects at planning stage. This leads to conclusion that in any part of data transport network the designed link utilization value should be kept under 30%.

16.2 Clock and Timing Network

Time synchronization and clock distribution network.

White Rabbit network. The detector electronics is connected to WR switches by 1 Gbps fiber-optical links. WR Switches are connected in a redundant, fault tolerant, 3-level fat tree topology. The top Level 1 is the Timing Master. Level 2 is a time distribution level. The 3rd level of WR switches connects to DRE boards. White Rabbit network has dedicated data connection to Front-end Control Network for low bandwidth (up to 0.5 Gbps) configuration data transfer.

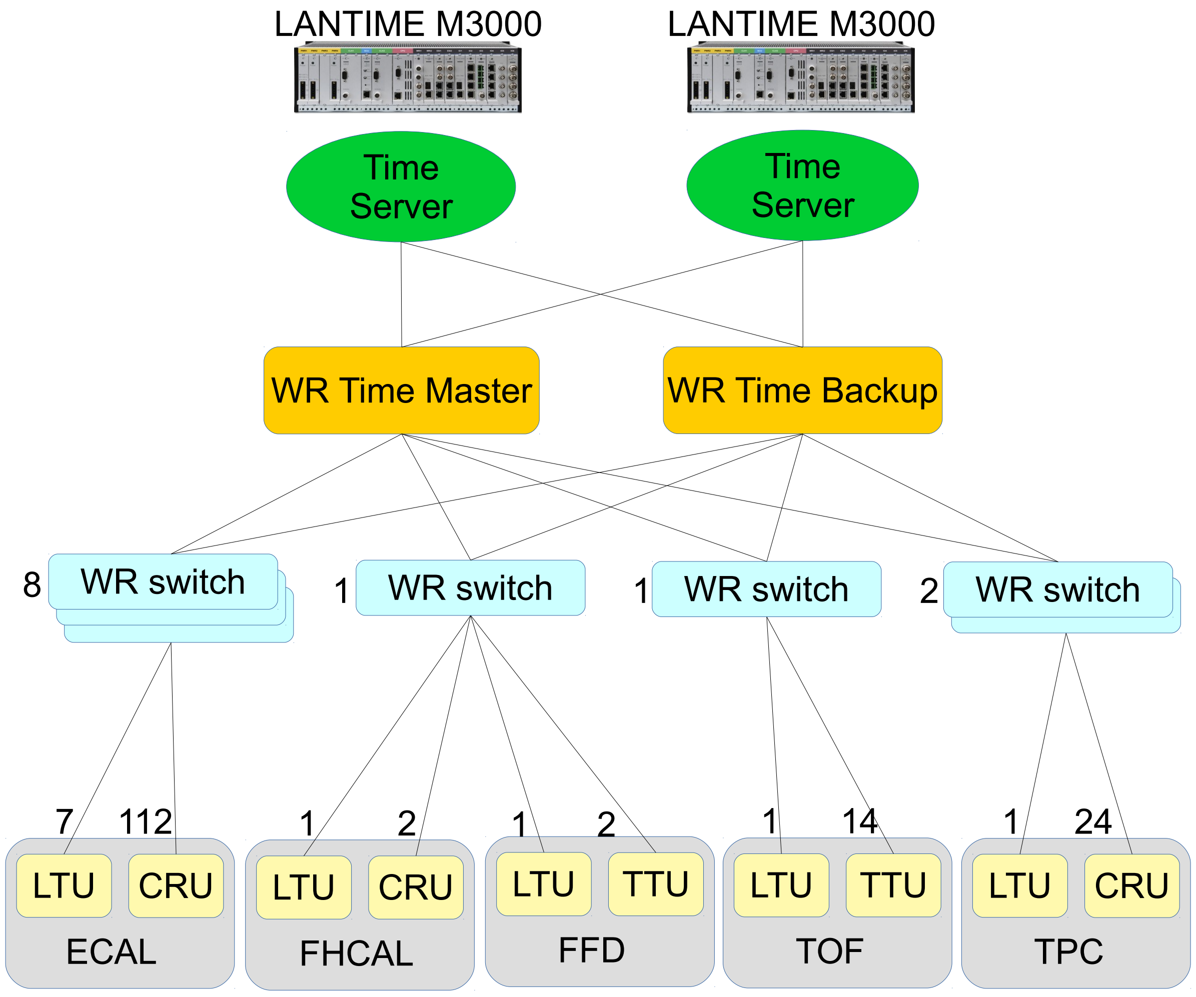


Figure 24: White Rabbit Network

The estimated WR link and switch count is summarized in Table 10.

Detector	Number of WR links	Number of WR switches
ECal/B	119	8
FFD, FHCal	6	1
TOF/B	15	1
TPC	25	2
Distribution switches	-	2
Total	165	14

Table 10: White Rabbit network connections

16.3 Trigger Distribution Network

16.4 Front-end Control Network

16.5 Online Storage Network

Online (Transient Data) Storage Network is internal cluster network that connects nodes of Transient Data Storage system. To provide storage service with specified I/O operation parameters it is required for this network to have minimum latency and sufficient bandwidth. The network should be capable of transferring replication data traffic that bandwidth is limited by storage device performance and node processing power. Solid-state storage devices that are used in TDS system have I/O bandwidth that may exceed network card capabilities and link saturation is expected in storage network. The packed drop caused by link saturation or network adapter queue overflow is normal and expected behaviour and is considered as a flow control.

Taking into account network saturation this cluster storage network should be isolated and should not be shared with online processing network that requires lossless operation with minimum packet delays.

Replication traffic in software defined storage network has horizontal pattern. The two-tier Spine-Leaf topology is a scalable and proven solution. Cut-through switching minimize packet delays and improves I/O latency.

The prototype of online storage system has been designed, built and put into operation in 2014. It has been used as online storage system for BM@N experiment. Few upgrades were performed in March 2018 the online storage network were operating with Cisco Nexus 5672 switches. The aggregate link bandwidth is 320 Gb/s. Few storage disk failures were observed and replication traffic was saturating existing network links.

16.6 Online Processing Network

Online Processing Network has main function of data transfer. It is built of commercially available networking equipment. This network connects together the equipment located in MPD electronics racks and parts of Online Computing System.

The bandwidth of the network fabric should be enough to pass the traffic between all parts of the DAQ system under operating conditions. No specific latency requirements are set for data network. The architecture of data network should be non-blocking and allow data flow without packet loss at projected data rate.

The same requirements of reliability are applied to both Detector Readout and Online networks. Whenever possible the MPD data acquisition process should be continuous. The network should be redundant and fault tolerant. No single point of failure is allowed for the network itself, it should continue to operate in case of component failure, possibly with reduced performance.

All the FLP, EvB, DR, HLT processing nodes therefore will be connected to Top-of-Rack (ToR) switches of the network fabric. ToR switches forms the leaf layer of the two tier Spine-and-Leaf topology. Every leaf switch in fabric is connected to each of the spine switches by fabric link. The fabric link bandwidth was first defined in this TDR as 40 Gb/s. As the commercially available Ethernet technologies rapidly evolve, first DAQ network fragment will have 100 Gb/s fabric links and 400 Gb/s Ethernet standard is considered as upgrade path after 2020 year.

16.7 DAQ Network Architecture

Taking into consideration the requirements that are set by the Detector Readout Network, the Online Storage Network and the Online Processing Network, the architecture of the DAQ Network has been designed. The long term experience that has been gained while operating DAQ networks of Nuclotron experimental setups (BM@N and others). These DAQ networks has been built with very limited set of hardware and their ad-hoc architectures were far from optimal - no redundancy, complexity of operation, insufficient bandwidth. There are several solutions to overcome these problems and clearly designed architecture is of paramount importance.

The DAQ Network has many common properties with well known data center networks, in contrast to campus networks. The traditional data center network uses a three-tier architecture. It consists of core routers, aggregation (or distribution) and access switches. The Spanning Tree Protocol (STP) is used to build loop-free topology for the OSI Layer 2 part of the network. STP is simple to setup, but cannot use parallel forwarding paths since it always blocks the redundant paths in a VLAN.

The multichassis link aggregation (MLAG) technology has been implemented by network equipment vendors to enable parallel path utilization. All redundant paths between core and aggregation, aggregation and access layers are forwarding traffic in MLAG enabled network. Spanning Tree is still used as a fail-safe mechanism.

The traditional three-level network design is well suited for north-south traffic pattern which may be observed in campus or traditional data center networks. With the evolution of virtualization technologies, the traffic patterns changed and in modern data center with high use of host and storage virtualization more traffic go in horizontal (east-west) direction, from one access switch to another.

A new Spine-and-Leaf architecture (Figure 25) design based on the Clos network overcomes the limitations of traditional three-tier network architecture. It has predictable packet latency and nonblocking connectivity between access endpoints.

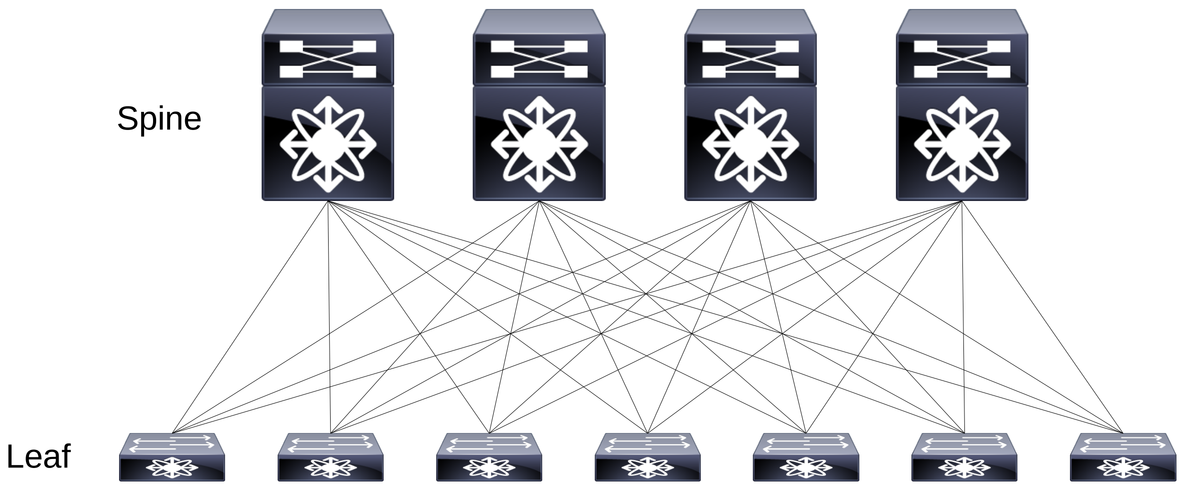


Figure 25: Spine and Leaf Topology

In this two-tier Clos architecture every access-tier (Leaf) switch is connected to each of the core-tier (Spine) switch in a full-mesh topology. The access endpoints are connected to the leaf switches. The spine switches form the backbone of the network. The traffic is distributed over parallel paths providing efficient link utilization and redundancy.

The Spine-and-Leaf topology is scalable and provides a straightforward way of expanding the bandwidth by adding more spine switches. The number of access ports may be simply increased by adding more leaf switches.

In Spine-and-Leaf architecture the packet latency if predictable and the traffic has to traverse fixed number of switches no matter of what is the source and destination endpoints within the fabric unless both endpoints are on the same switch.

16.8 DAQ Network Virtualization

The large network for the virtualized environment must meet certain requirements. For example, the network must provide Layer-2 connectivity between hosts in the virtualization cluster to enable virtual host migration and service high availability with redundancy protocols like VRRP. The network should be reliable and fault tolerant and provide segmentation between failure domains. Network must meet needs of scaling of forwarding tables and network segments. The essential requirement of network for physical experiment is device mobility. On the other side, all these requirements must be met using shared physical infrastructure and the network should be controlled as a single entity by DAQ network operators.

The network virtualization technologies and software defined networking are used to build the architecture that meets the above requirements.

The network virtualization is based on the concepts of Overlay and Underlay networks. The underlay network provides connectivity between network devices within the fabric, and the overlay network carries the endpoint (payload) traffic. The overlay logical topology is decoupled and independent from the underlay topology. Overlay traffic is carried over underlay network in encapsulated form as a black box. The same network fabric may carry traffic for multiple independent and isolated overlay networks that facilitates application and tenant segmentation. The overlay encapsulation allows the address spaces to be administrated independently from the underlay network and allow to use same MAC and IP addresses in different overlay networks. The address space independence is becoming important when deploying the same software application by multiple tenants or when multicast technology is used.

Software-defined networking (SDN) technology facilitates the network management, configuration and monitoring. In contrast to traditional, static network architecture, SDN architecture enables to programmatically adapt network configuration to the dynamically changing application requirements. The SDN architecture is centrally managed as a single entity by software based SDN controllers that maintain the configuration of the network fabric.

The virtualization and SDN solutions well match the requirements of DAQ Networks. Many network equipment vendors started to provide SDN solutions. In the Gartner report from July 2017 were defined two leaders - Cisco and Arista Networks - that have the most complete vision and ability to execute in Data Center Networking. After analyzing the SDN solutions from network equipment vendors and taking into account our positive experience with Cisco Nexus switches and technologies the Cisco SDN solution has been chosen for DAQ network architecture.

16.8.1 BMN Online Processing Network

The existing DAQ network at BMN experiment (Figure 26) has ad-hoc, specific architecture that has to be upgraded to satisfy the needs of upcoming experimental run in 2019 (Figure 27).

The hardware of the BMN DAQ network is mostly Cisco Nexus switches.

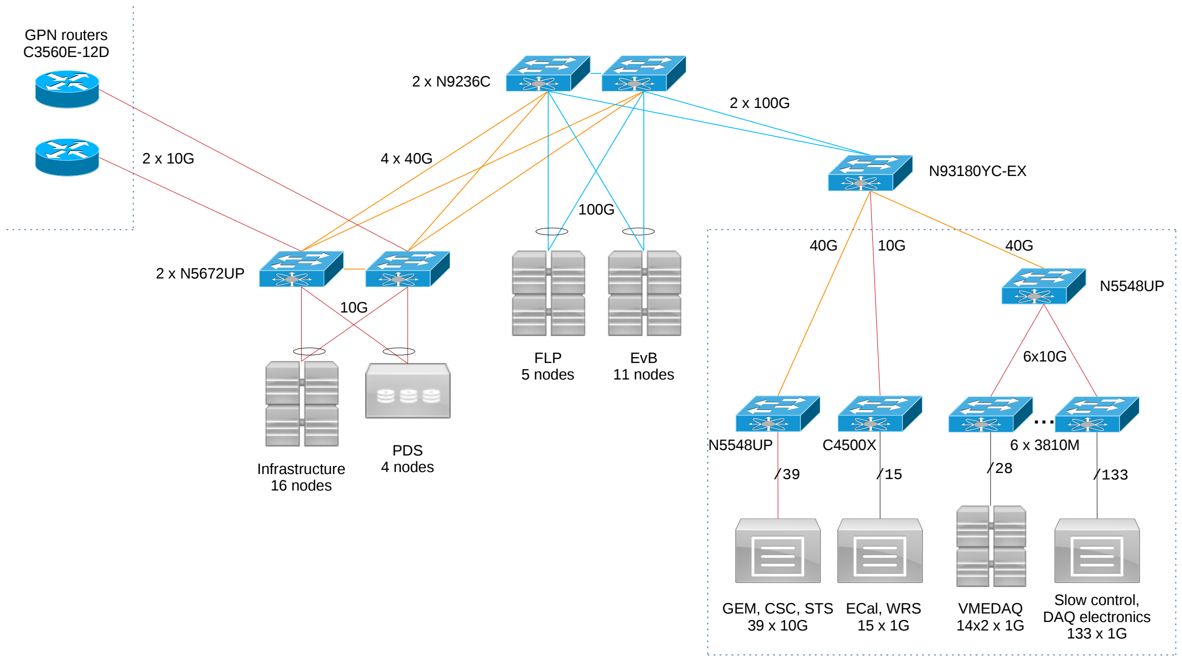


Figure 26: BMN DAQ Network in Run 2018

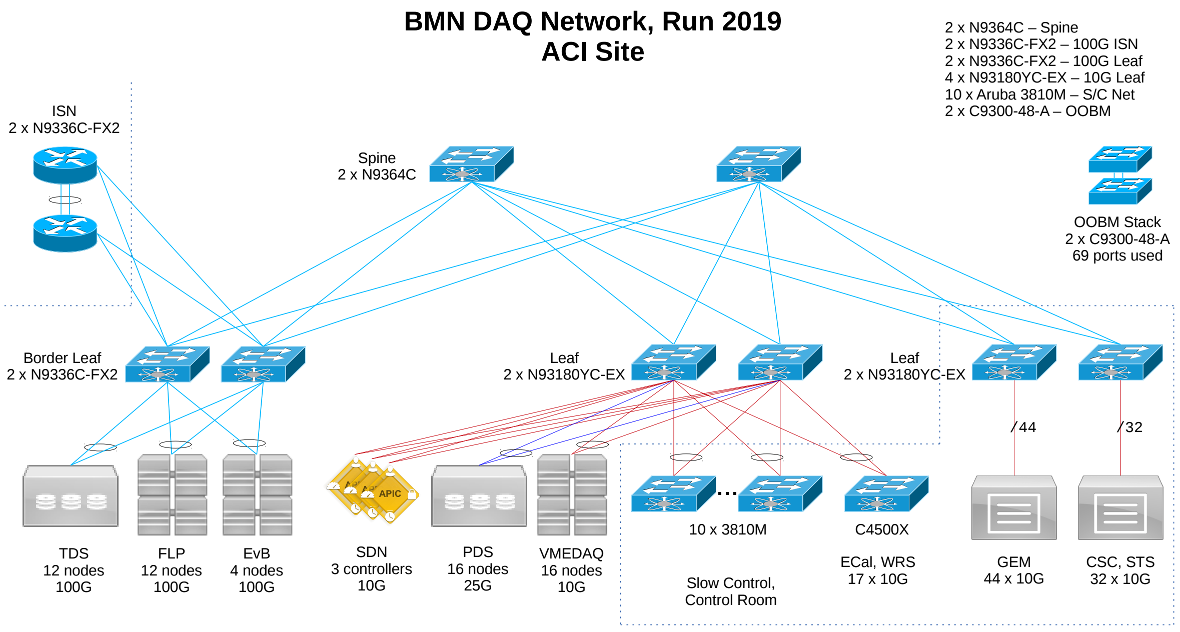


Figure 27: BMN DAQ Network for Run 2019

16.8.2 MPD Online Processing Network

The deployment of MPD DAQ Online Processing Network is performed by stages. In first stage, planned for the end of 2018, the network core is deployed. It will allow to start operation of basic fragment of MPD DAQ data processing pipeline and it will allow the operation of MPD subdetector assembly and test facilities in building 42 and MPD DAQ electronics test facility in building 201. The diagram of first stage of MPD DAQ Network is shown on Figure 28.

The MPD DAQ Online Processing Network is built with Cisco Application Centric Infrastructure (ACI) technology. The network topology is Spine-and-Leaf Fabric. Three SDN controllers (Cisco Application Policy Infrastructure Controller, APIC) forms minimal cluster that is necessary for the operation of SDN network.

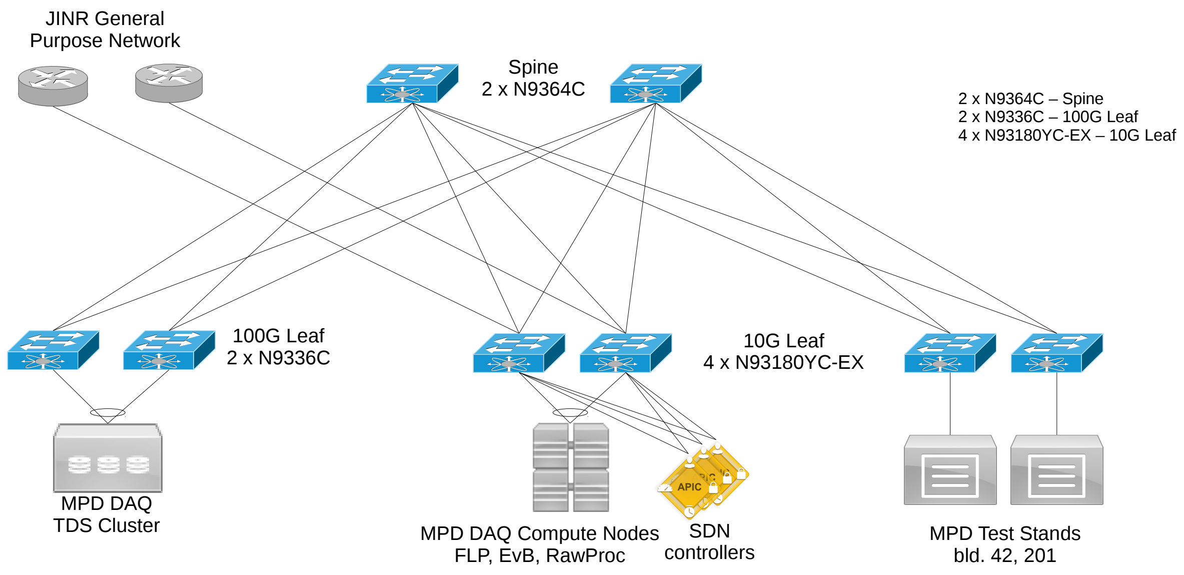


Figure 28: MPD DAQ Network Diagram (ACI Fabric) for 2018

16.9 BMN and MPD DAQ Network

The conceptual design of the BM@N and MPD DAQ networks is presented on Figure 29. The design is based on Cisco ACI Multi-Site technology. It allows to interconnect multiple ACI cluster domains (fabrics), each representing a different availability zone.

This design is optimized in terms of cost and it matches both the BMN and MPD DAQ Network requirements. Four ACI Fabrics for BMN and DAQ computer centers and experimental zones, each at different location, are interconnected with Inter-Site Network (ISN) in full-mesh fashion that allows to minimize number of long-distance fiber links for ecomonical reasons.

The given design is universal, it may scale horizontally to match the requirement of increasing number of connected endpoint devices by adding more leaf switches as well as increase end-to-end throughput by adding more spine switches. SDN controllers (APIC) are distributed over sites for high availability.

The Out-Of-Band Management (OOBM) switches form dedicated network (not shown on diagram) that is necessary for control of the network.

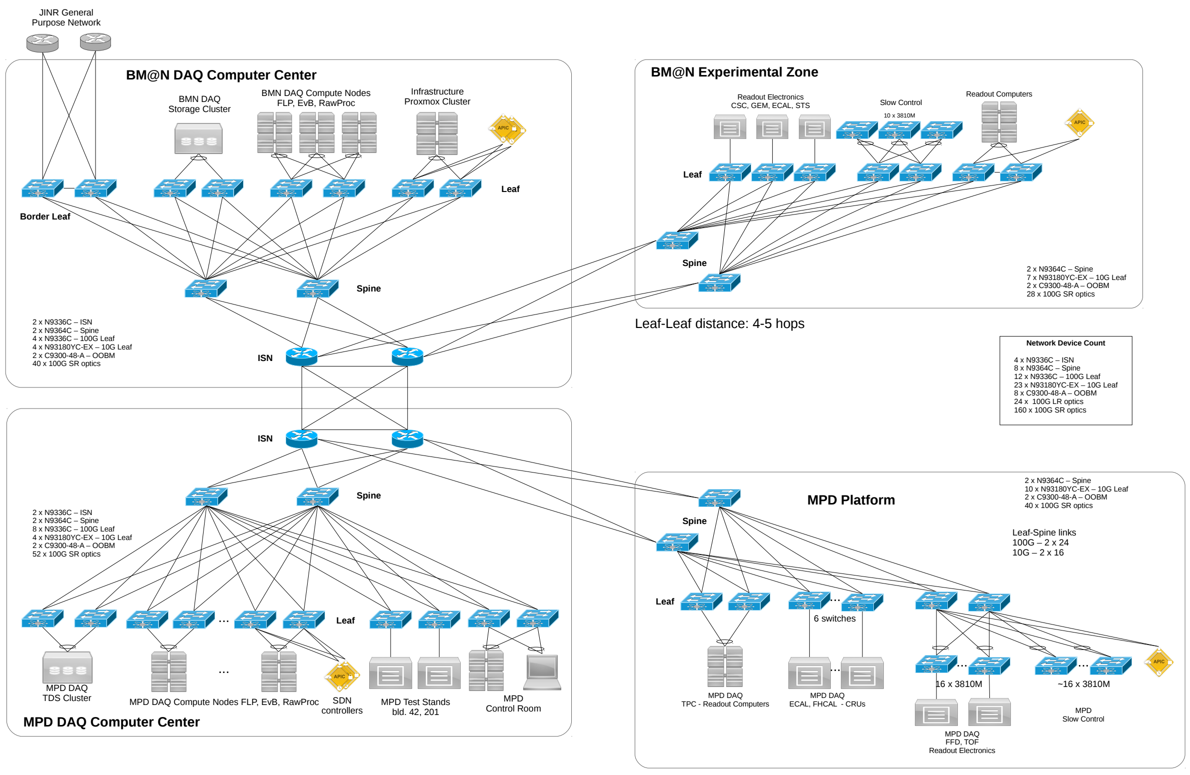


Figure 29: DAQ Network Diagram (ACI Milti-Site)

17 Data Storage System

The Storage System is the final step of the MPD data flow. All the events selected by the trigger system are archived for further analysis.

17.1 Storage Tiers

The MPD Storage System has two stages, the Transient Data Storage (TDS) and Permanent Data Storage (PDS). It utilize two-tier software architecture and data migration from TDS and PDS is transparent to the user. The storage system exposes standard interfaces to the applications. POSIX compatible cluster file system as well as raw Object Storage are available.

The Transient Data Storage (TDS) will perform the temporary storage of raw data produced by Event Builder system. It is a low latency and high throughput soft real-time distributed cluster storage system based on solid state memory devices and cluster software. It will allow the independent functioning of the MPD DAQ for a period of 24 hours in case of software, hardware, network or permanent storage outage or maintenance. The data are recorded to the TDS during data-taking phase of MPD and is continuously migrated to permanent storage.

17.2 Data Redundancy

The important requirement for storage system is the ability to retain the data for long period of time without corruption or loss. The storage elements are not ideal, hard disks failure rate is about 1 percent per year for enterprise class devices. To mitigate the possible disk drive failures the redundancy is used. Multiple copies of same data are stored in distinct servers and racks. The algorithms differ from simple replication to Ceph Erasure Coded Pools[7]. The replication is straight to implement but requires more raw space, while erasure coded pools are computational intensive, require times more processor power and are tricky to recover in case of failure.

Since the reliability of the non-ideal system is always less that 1.0, the required system reliability should be defined as a target. There are many other causes of storage system failures besides the storage elements itself. Taking into account only storage elements (hard disk drives) reliability, and the expected probability of single bit of data loss 10^{-6} per year, it is necessary to use triple replication or triple raw storage capacity. The erasure coded pool (10,4) RS will have better reliability, only 40 percent capacity overhead by the price of software complexity, CPU time usage and network bandwidth.

17.3 Storage Capacity

The amount of data stored is calculated in assumption of NICA effective working time coefficient 0.9, trigger rate $7kHz$. During one day of operation MPD will produce 6×10^8 raw data events. Assuming raw data event size of 1000 kB it require 500 TB of space in Transient Storage System. It is still possible to store raw data without reduction for short data taking runs: 10 days of operation will require about 5 PB of storage space. It is evident that for physics data taking run further data size reduction is necessary to keep storage space requirement under limitations of Permanent Storage System that are dictated by economical and infrastructure reasons.

The task of data reduction by running software algorithms is handled by High Level Trigger.

This process has no impact to the data taking process and does not throttle the trigger. The data is being read by HLT from Transient Storage, processed by independently running processes on the HLT Nodes and resulting data is written back to TDS.

The HLT process increases space requirements to the TDS system by 10 to 20 % depending on the compression ratio achieved. The I/O operation number is increased by 200% (besides raw data write operation an extra read and write operation is added).

For ECAL subdetector the pulse feature extraction is performed by running peak parameter estimation in software. The expected reduction ratio is near 1:10.

For TPC subdetector the lossless compression algorithms using Huffman and Arithmetic Coding may reduce data size by 60%. The lossy Vector Quantizer may achieve ration of $\sim$ 50% with a measurable but small impact on the space point resolution. By using online pattern recognition hte TPC data can be modeled more efficiently by representing the data using cluster and track information. The results show compression ratio of 10-15 % [8]. The computational resource required to run highly optimized algorithms is in order of 1 second per event per one processor core of Sandy Bridge architecture. At the rate of 7000 Hz this will translate to 350 20-core processor nodes in 2015 or 200 processor nodes in 2017.

To estimate the storage space requirement it is assumed the overall raw data compression factor to be in range 10-20% (compression ratio 1:10 to 1:5). This implies that original raw data from front-end electronics is discarded and only compressed data is stored permanently.

The estimated requirements for transient storage space are summarized in subsection 17.3.

Parameter	Value
Beam	Au+Au 9 GeV
Trigger rate	7 kHz
Event size	1000 KB
Raw data rate	6.5 GB/s
Beam availability factor	90 %
Daily raw data volume	500 TB

17.4 Data Storage Technology

To evaluate storage technologies and performance the fraction of MPD storage system has been built. It is comprised of commodity storage servers with hard drives with spinning platters as main storage media and high performance solid state storage used as journal. The Ceph[7] software cluster provided the distributed, software defined storage system with file level interface. It has no single point of failure and by design is self-healing and self-managing. By using data replication it is fault tolerant.

The essential part of storage system is data transfer network. The links interconnecting storage nodes have bandwidth enough for data transfer to and from the storage system itself and for internal data replication process.

The Ceph cluster consists of three kinds of nodes. Monitors keep track of active and failed cluster nodes. Metadata servers store the inode and directory information. Object storage devices actually store the content of files.

The performance of Ceph storage cluster is defined by hardware configuration, ceph and system software versions and Ceph tuning parameters. The Ceph software may accommodate to different setups and will run on commodity server hardware.

It is critical to ensure that every element in data transport path has enough bandwidth to sustain design data throughput rate and every I/O operation is completed in reasonable time.

The hardware setup of storage node has been designed using experimental results of CERN-IT[] and Inktank [].

The data flows inside object storage node estimated for sustained maximum throughput are shown on Figure 30. Numbers represent maximum possible throughput limited by storage devices and bus interconnects. The figure represents BM@N storage node hardware.

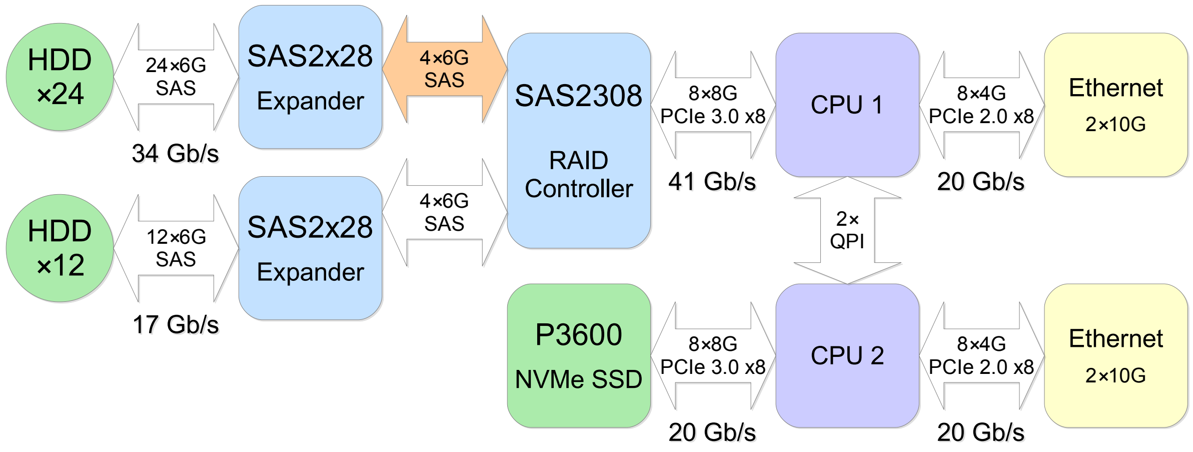


Figure 30: BM@N Storage node bus data flow

17.5 Transient Data Storage Hardware

TDS Tier is a part of Online DAQ system that has direct impact on data acquisition process. It should sustain the data flow from the event builders and the I/O operations should complete in near real time thus minimizing trigger throttling. TDS is an intermediate data buffer for all online systems.

The reliability if TDS is of high concern. Since it is a distributed and fault tolerant system, the process of error recovery after failures is considered normal operation and the TDS system should meet the design parameters is any operation state.

The Ceph data recovery process triggered by hardware failure, network disruption or software malfunction causes significant increase of I/O rate on storage devices and increase of network data transfer. The Ceph parameters limits the data recovery speed to allow enough resources to primary data storage function.

The TDS Tier storage capacity is not critical, but the performance in terms of operations per second (IOPS) and latency is very important. Hard disks with high speed rotating platters (15000 rpm) were used for this purpose for years, but nowadays the Solid State Storage based on Flash memory significantly outperforms spinning hard disks in IOPS, latency, reliability, power consumption and size.

We have considered the Flash storage elements in 2.5” hard disk compatibility form-factor with SATA or SAS interface and NVM-Express (NVMe) storage cards in PCI-Express format. The NVMe cards have better performance characteristic since the interface does not use intermediate SATA or SCSI layer and software talks directly to storage hardware.

The configuration of object storage node hardware for TDS prototype used in BM@N data taking runs is shown in Table 14.

The configuration of TDS hardware for MPD DAQ is under consideration. The preliminary configuration of storage nodes is shown in Table 11.

Taking into account that TDS must contain raw data for 24 hours of MPD operation, and the required useable data volume 500 TB, the storage structure has been designed. The parameters of TDS system are shown in Table 12

System	Ultra SuperServer 1029U-TN10RT
CPU	2x Intel® Xeon® Platinum 8168 Processors
RAM	12x 32GB DDR4-2666 RDIMM
Object Storage	10x 2.5” 6.4TB Micron 9200 MAX NVMe SSDs
System disk	1x 240GB Micron 5100 PRO SATA M.2
Network	Dual 100GbE QSFP28 via 2x Mellanox MCX516A-CCAT
Chassis	1U with redundant Hot-swap 1000W power supplies

Table 11: TDS storage node configuration, 2018

	BM@N-2018	MPD TDS
Useable space	110 TB	512 TB
Replication type	3-replication	2-replication
Drive type	4 TB SAS	6.4 TB NVMe
Drive capacity	3.7 TB	5.8 TB
System-full watermark	80%	90%
Data redundancy overhead	200%	100%
Drive bays per server	36	10
Number of storage servers	4	22
Total drive count	110	220
Total number of network connections	16 x 10 Gb/s	44 x 100 Gb/s
Number of storage switches	2	2
Rack space usage	20 U	26 U
Power usage	3.5 kW	14 kW

Table 12: TDS parameters

17.6 Permanent Data Storage Hardware

PDS Tier should have stable sustained throughput for multiple concurrent read and write access. The latency is not a limiting factor since all operations on PDS are decoupled from TDS and thus have no impact on detector data readout process. From the other side, read latency should be kept under 1 second to allow interactive operations on the data performed by data analysis software.

Common requirements for PDS hardware are performance, reliability and cost efficiency, including both materials and operational expenses.

The configuration of object storage node hardware dedicated for PDS is shown in Table 13.

Motherboard	Supermicro X9DRi-LN4F+
CPU	Dual Intel Xeon E5-2650v2 (8 cores, 2.6 GHz)
RAM	64 GB DDR3 ECC
Object Storage	30 × 4 TB Seagate ES.3 SAS-6G HDD
Journal SSD	Intel P3600 2 TB NVM Express Solid State Storage
System Disk	2 × Intel DC3500 240 GB SSD in software RAID-1
Network	2 × Dual 10 Gb Intel x520
RAID Controller	LSI 2108 SAS-6G, PCIe 2.0
SAS Expanders	2 × LSI SAS2X28 SAS-6G

Table 13: Object storage node hardware setup

17.7 Compute Hardware

To estimate the processing performance of compute nodes that will be available at the time of hardware order the data of existing supercomputer systems were analysed. From the TOP-500 supercomputer data [9] it is found that the performance expressed in GFLOPS (Giga Floating Point Operations Per Second) is increased every years, however the rate of this increase is not a constant. In the period of 1995 - 2010 the GFLOPS were increased by 1.91 times every year and in 2010 - 2016 the rate was 1.47. I one take only 2012 - 2016 period data the increase rate is even lower, 1.40. By extrapolating the last number it is expected that under current technological conditions the performance will increase by 3.8 times in next 4 years (2016 - 2020). See Figure 31.

The configuration of object storage node hardware for TDS is shown in Table 14.

Motherboard	Supermicro X9DRi-LN4F+
CPU	Dual E5-2690v2 (10 cores, 3.0 GHz)
RAM	128 GB DDR3 ECC
Object Storage	3 × Intel DC3700 400 GB SSD
System Disk	2 × Intel DC3500 240 GB SSD in software RAID-1
Network	Dual 10 Gb Intel x520
RAID Controller	LSI 3108 SAS-12G, PCIe 3.0

Table 14: Compute node hardware setup

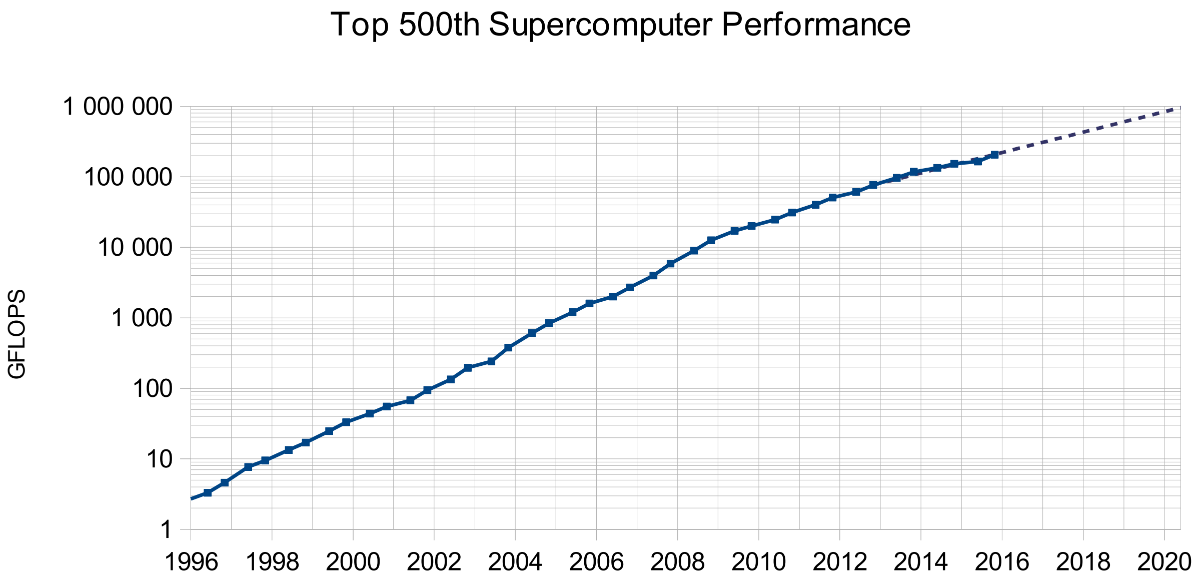


Figure 31: Top 500th Supercomputer Floating Point Performance

17.8 Benchmarks

17.8.1 Test Setup

Number of benchmarks and reliability tests were performed on PDS Test Cluster. The cluster consisted of 3 storage nodes (see Table 13). The Ceph monitors run on same hardware.

Every node is connected by dual 10 Gb/s links to pair of nonblocking network switches Cisco Nexus 5548. Dual links are aggregated into virtual Port-Channel. In this configuration the single flow throughput is 10 GB/s. Multiple flows are balanced between two links in Port-Channel and for high number of flows the total throughput is near 20 Gb/s.

17.8.2 Write Test

The performance measurements were performed on idle system at object storage level with internal Ceph benchmark tool. The workload was generated by single client performing multi-threaded write to object pool. By varying the number of concurrent writes and the block size the total throughput and I/O latency were measured. Limiting factor is network bandwidth thus the total bandwidth is below 1200 MB/s for triple replication.

Test results are shown on Figure 32 and Figure 33. Optimal I/O size is found to be 4 MB.

17.8.3 Long Time Performance Test

One hour of continuous write at maximum available throughput was performed and storage system performance was analyzed.

This test saturated storage system and exposed it's weak points. It is clearly visible at 17:35 (20 minutes of run time) that load balancing is not perfect and client write latency rises as high as 3 seconds for some I/Os. It is still acceptable for PDS Tier. The histogram of latency distribution is shown on Figure 34 and time series data is shown on Figure 35.

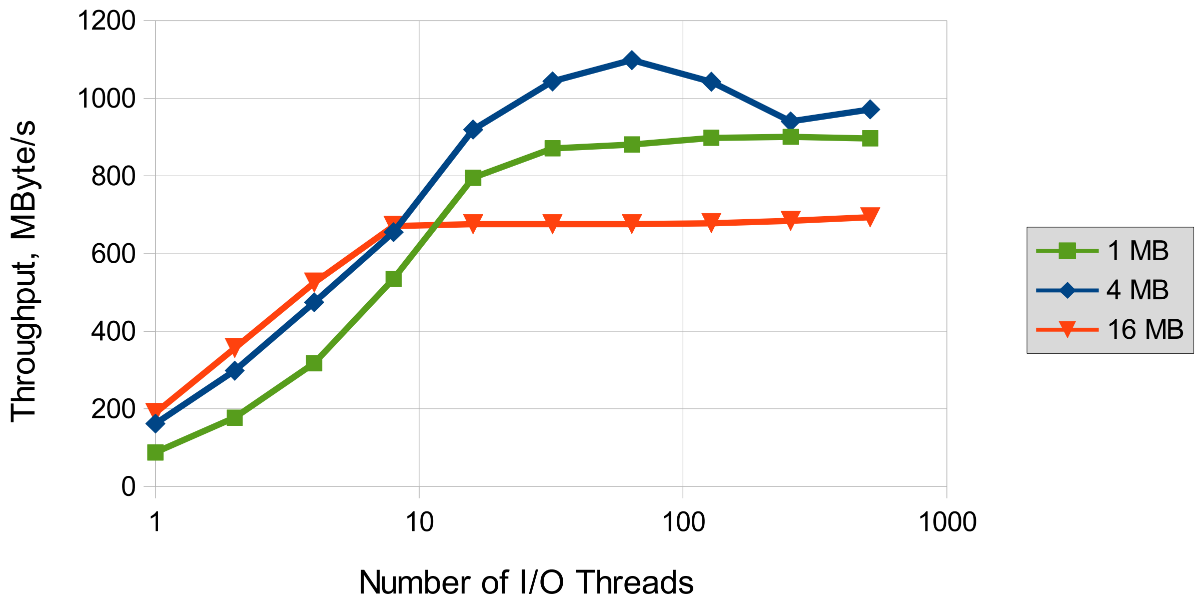


Figure 32: PDS-Test 3-node Ceph cluster object storage write throughput for different block size and number of concurrent operations

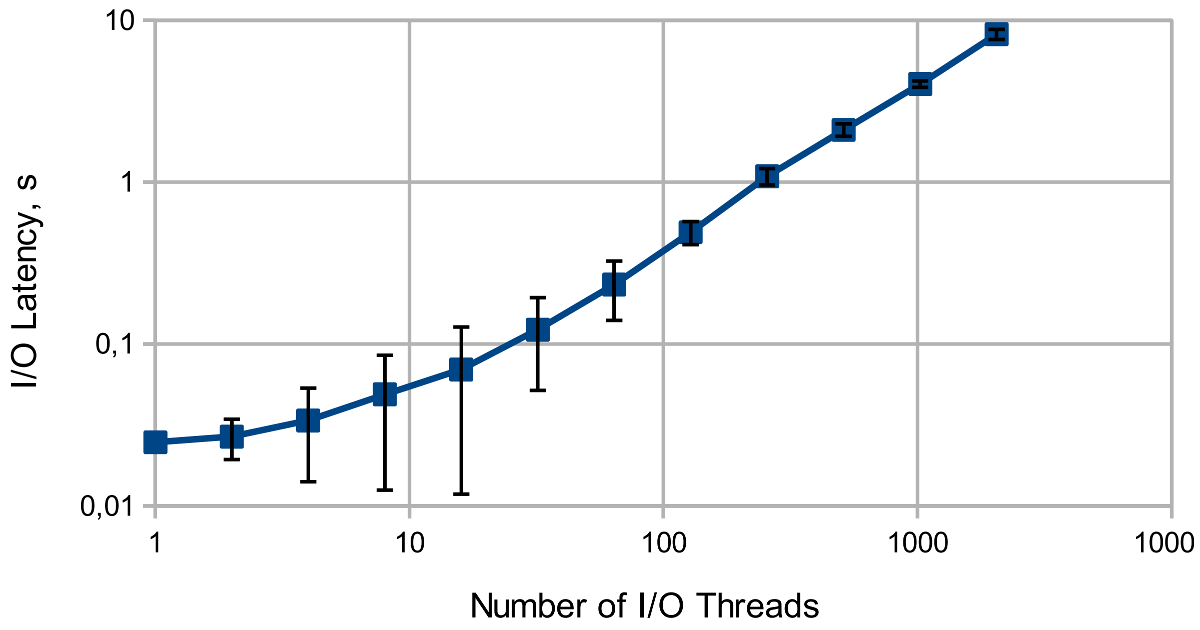


Figure 33: PDS-Test 3-node Ceph cluster object storage I/O latency for different number of concurrent operations

It is evident that the network bandwidth is limiting factor. As long as the test cluster is only a small fragment of complete system, and from experimental results from CERN-IT [] it is known that Ceph storage system scales out well by adding more object storage nodes, it is expected for complete PDS Tier to reach design parameters.

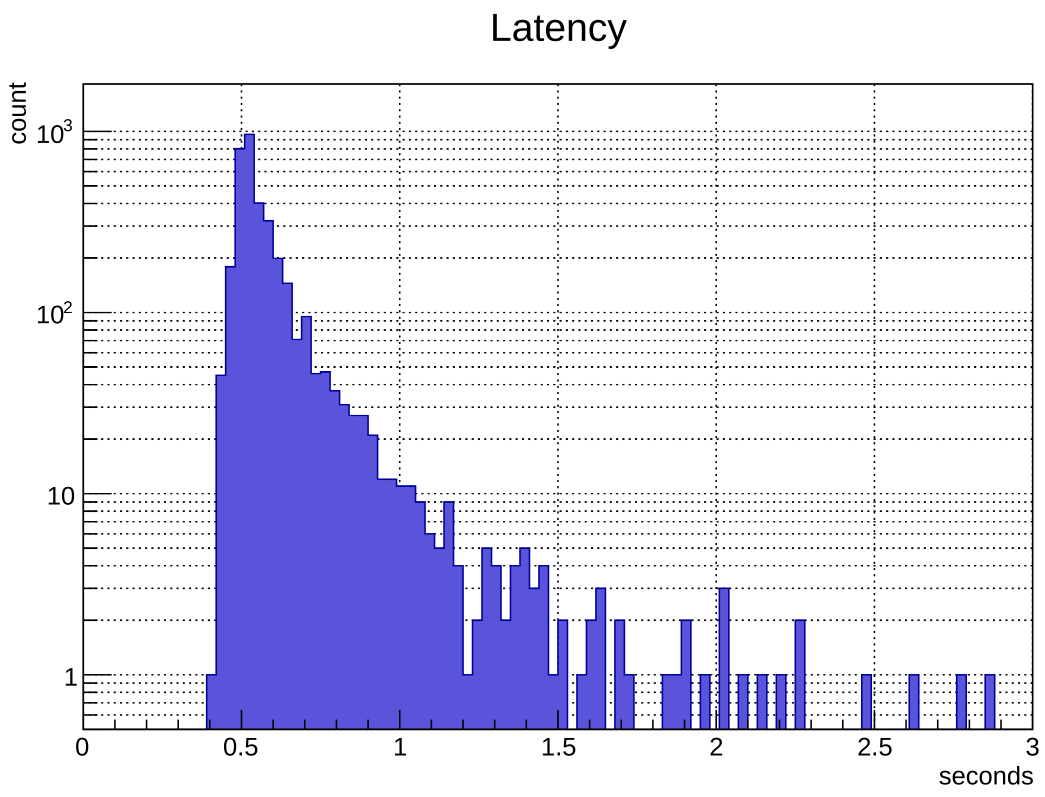


Figure 34: PDS-Test 3-node Ceph cluster object storage one hour continuous write - client side latency statistical distribution

Test results are summarized in Table 15. Compared to previous test run performed in 2014 on same hardware and Ceph version 0.87 Giant, the throughput increased by 65% from 540 MiB/s to 890 MiB/s, average latency decreased by 37%. For 99% of operations (99th percentile) latency is below 1.4 seconds (15% improvement). This shows significance of software optimizations and Ceph parameters tuning that were performed by Ceph project contributors.

Number of concurrent write operations	128	
Block size	4 MiB	
Test duration	15.5 hours	1 hour
Average I/O latency	0.95 s	0.6 s
Maximum (99th percentile) I/O latency	1.65 s	1.40 s
Average throughput	540 MiB/s	890 MiB/s
Ceph version	0.87 Giant	0.94.5 Hammer

Table 15: Object storage performance test results

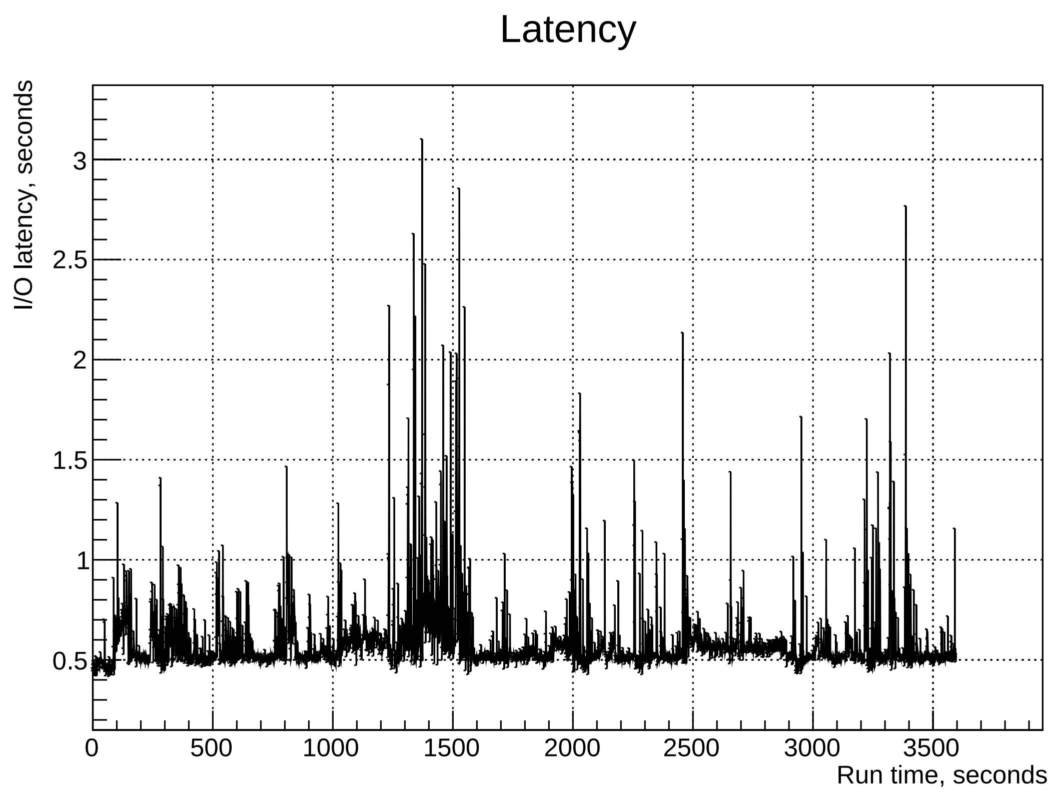


Figure 35: PDS-Test 3-node Ceph cluster object storage one hour continuous write - client side latency time series

17.8.4 PDS Performance Monitoring

PDS-Test Ceph cluster parameters are continuously monitored by Zabbix [10] monitoring system. Two main parameters: commit and apply latencies provided by Ceph cluster statistics for storage processes osd.0 to osd.49 are shown on Figure 36 and Figure 37.

Data is sampled every 30 seconds. Monitoring system has alarm capability based on thresholds. When monitored value meets threshold condition the alarm is activated and shown on monitoring dashboard. Optionally operator is informed by e-mail and SMS.

The monitoring system is operational since 2012 for JINR LHEP computing and network services.

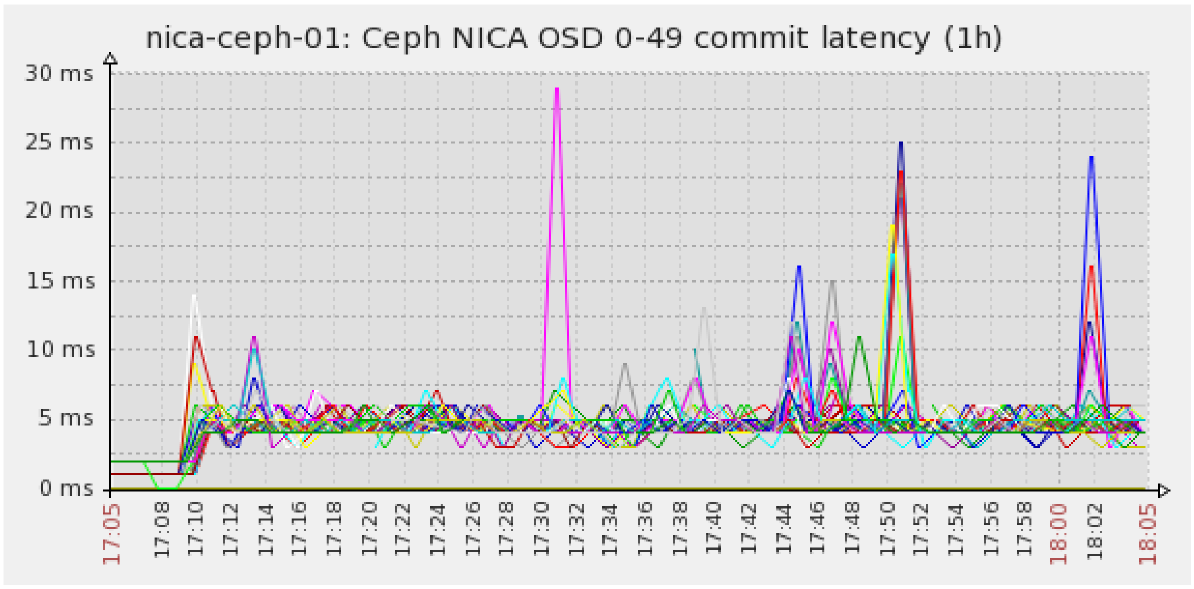


Figure 36: PDS-Test 3-node Ceph cluster object storage one hour continuous write - commit latency

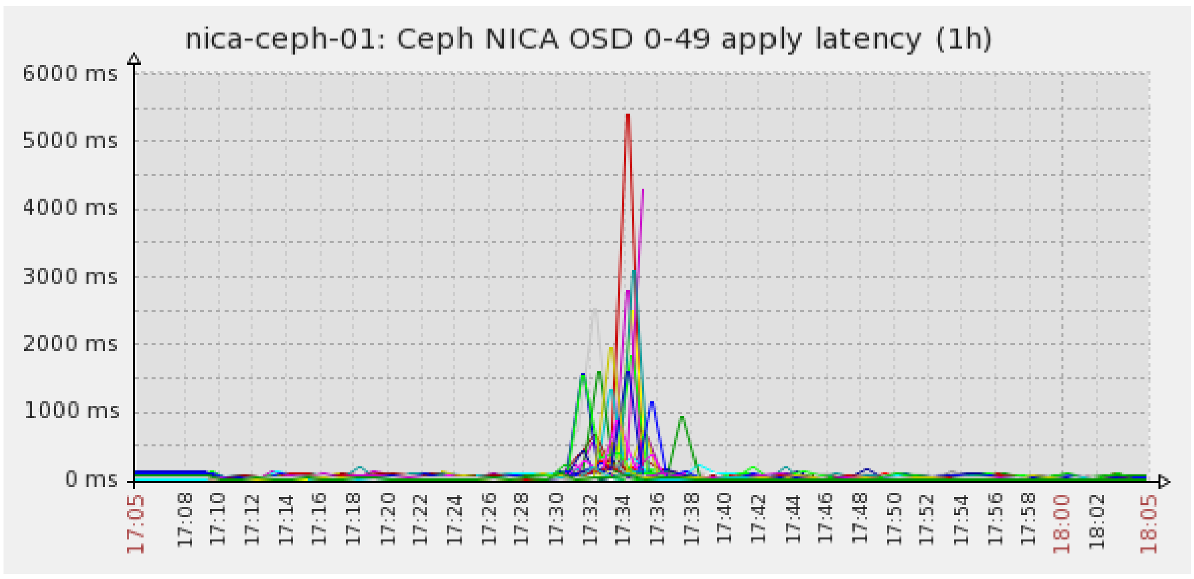


Figure 37: PDS-Test 3-node Ceph cluster object storage one hour continuous write - apply latency

17.9 Data Taking Run Results

The storage system was put in operation in November 2014 and was used as a primary storage system for BM@N fixed target experiment on Nuclotron. The object storage itself has not experienced any significant problems since all operation period, however the clients using the Ceph file system received interruptions in communication with Ceph cluster. The problem was tracked down and fixed by upgrading operating system on clients.

18 BM@N DAQ

18.1 BM@N Data Flow

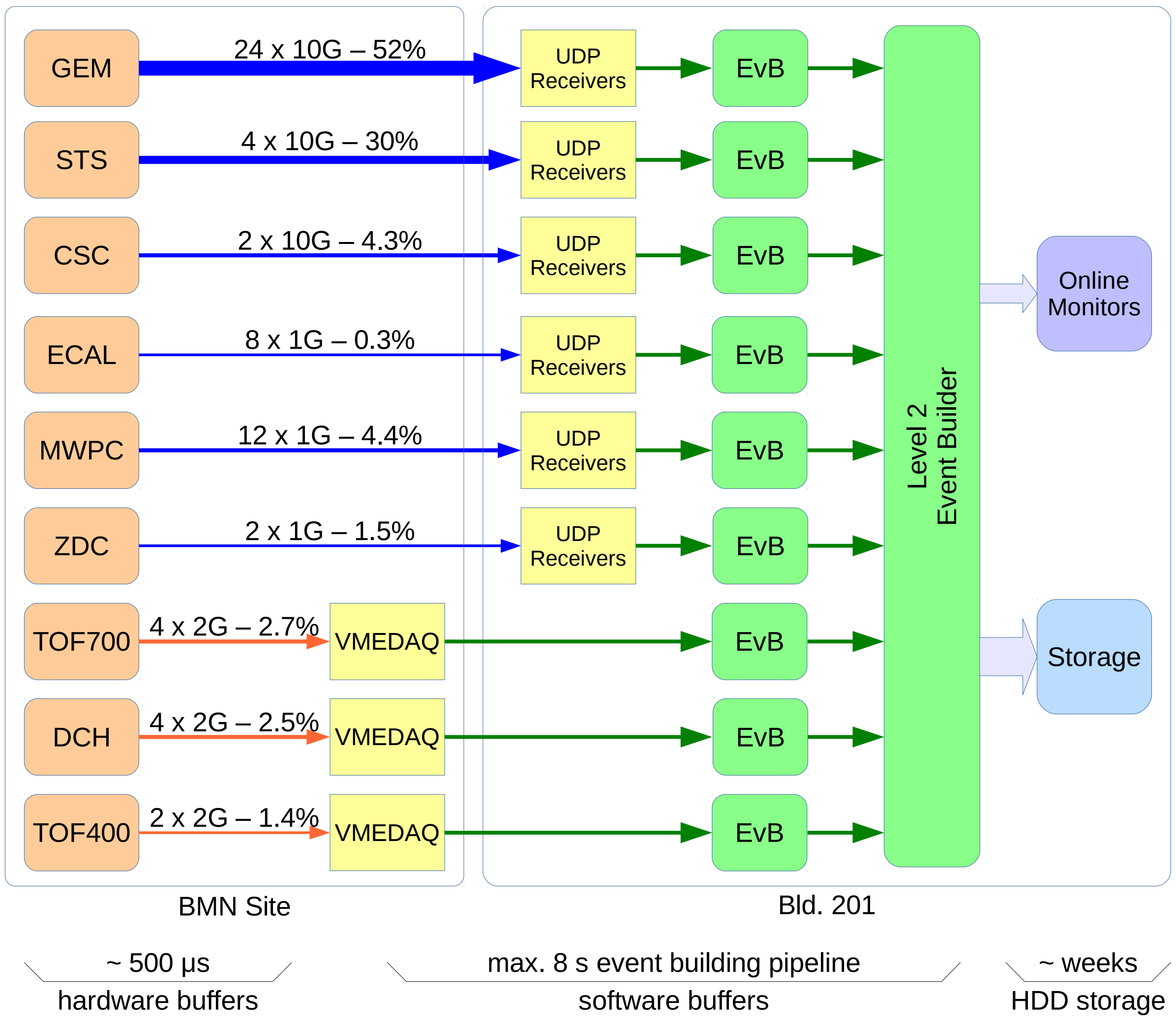


Figure 38: BM@N Data Flow Diagram

18.2 BM@N Clock Distribution

All Data Readout Electronics (DRE) in subsystems DCH, TOF700, TOF400, MWPC, ECal and FHCal have synchronous clocks. There are two ways, which used for clock synchronization: direct clock applying to the DRE and using recovered clock form WR link on the DRE boards.

DCH, TOF400 and TOF700 are VME based subsystems. FVME TMWR modules recover WR clock and adjust their own onboard oscillator to the same frequency and phase, then transmit this clock to the DRE modules by TTC Bus within one VME crate. DRE of ECal and FHCal are placed in close proximity to the detectors. All these modules are connected to the WR links and use WR recovered clock for synchronization with other subsystems.

MWPC's DRE have mixed clock distribution network. FVME TMWR module synchronized via WR network, transmit clock to TTB9 by TTC Bus, and TTB9 modules retransmit this clock to all DRE by HRB TTC protocol.

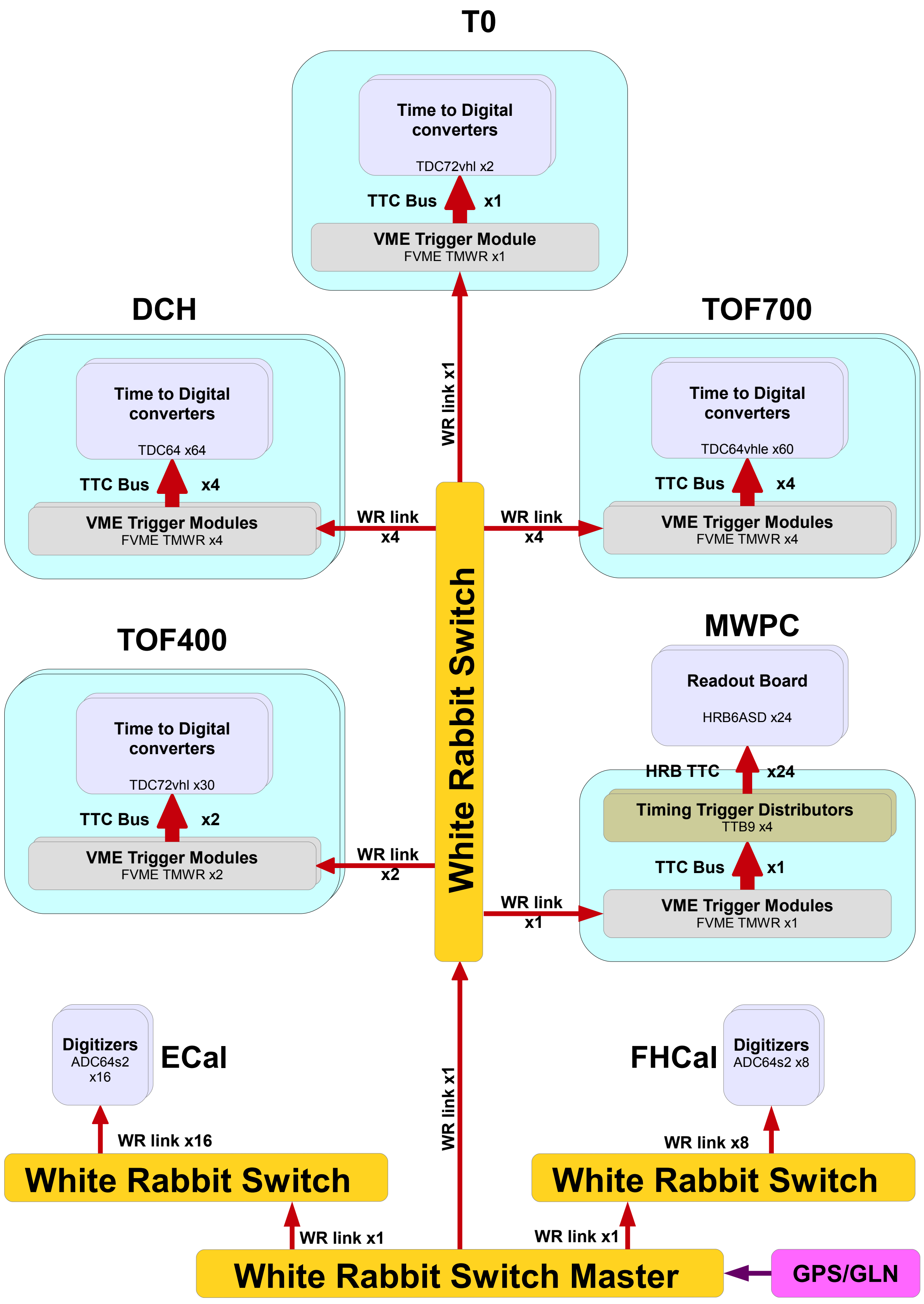


Figure 39: BM@N Clock Distribution

18.3 BM@N Global Trigger Distribution

T0 Unit generates Trigger and Spill signals and transmits them to TrigWord module, which returns aggregate XOFF signal form all subsystem electronics. This signal determine the Dead Time of the DAQ. The type of trigger configuration is defined by LVDS TrigWord Bus.

Global Run Control (GTU) module receives trigger signals form TrigWord module, distributes them via fanouts to Run Control Master modules (LTU) of each subsystems, aggregates all XOFF signals from LTUs and sends aggregated XOFF to TrigWord module. LTU provides trigger signals distribution in own subsystem.

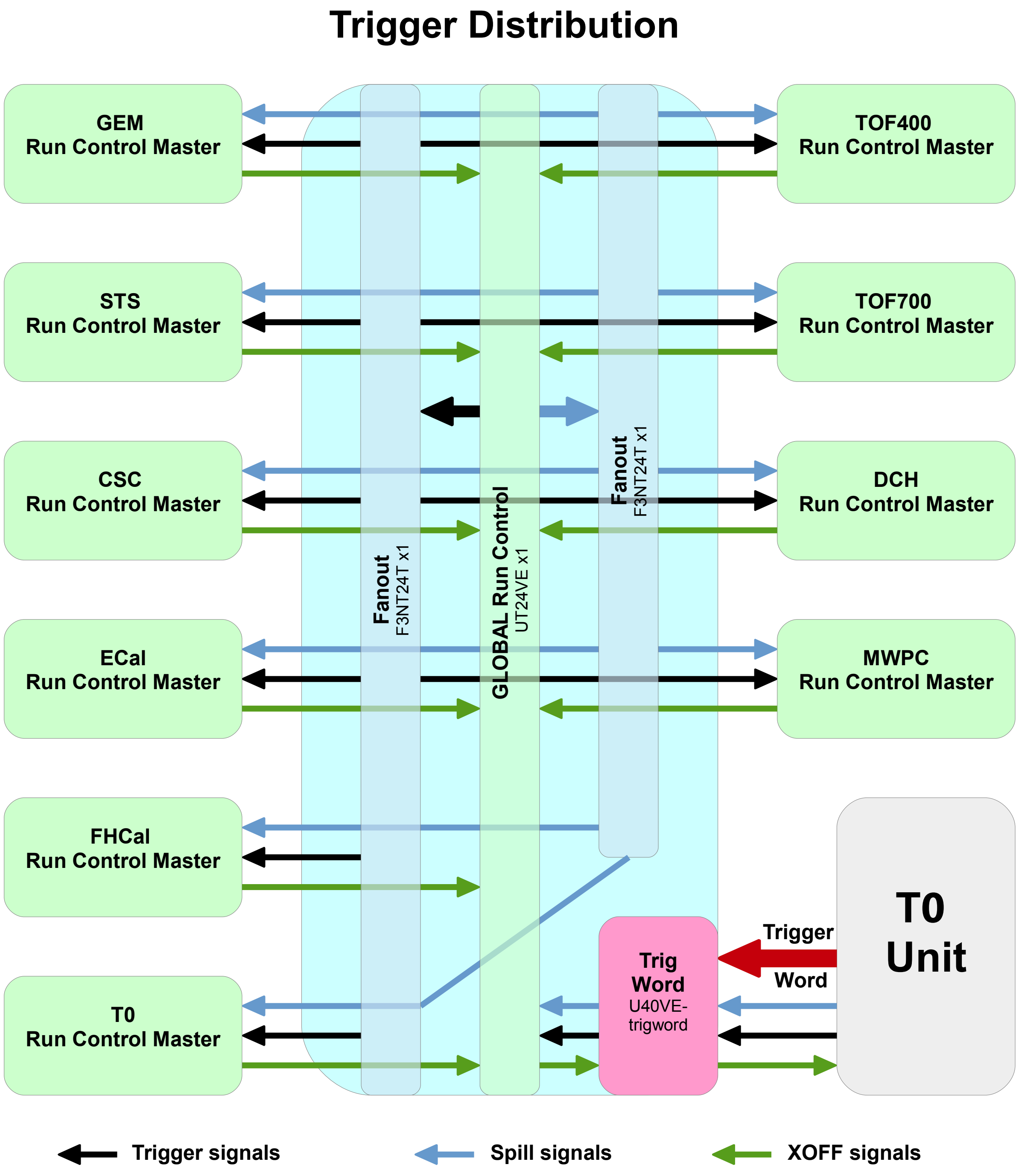


Figure 40: BM@N Global Trigger Distribution

18.4 BM@N Subsystems Trigger Distribution

Trigger distribution in Subsystems, which based on one type of the Digitizers (ADC, TDC), has similar architecture. There are two large groups: ADC-based and TDC-based systems. DRE modules in these group are installed into VME crates.

In addition, we can distinguish two another group: ECal and FHCal subsystems, and MWPC. DRE modules are standalone and installed onto detectors. This case requires different trigger distribution.

18.4.1 TDC-based subsystems

TOF400, TOF700 and DCH are TDC based subsystems and they have fully identical Trigger Distributing. The difference between these systems is various types of TDC modules. DCH has 64 channels 100 ps resolution TDC modules, TOF400 and TOF700 have 72 and 64 channels 25 ps resolution TDC modules respectively. All Trigger Distribution and DRE modules are installed into VME crate for powering, cooling and data transferring over VME 64x backplane.

The Run Control Master modules with fanout distributes trigger signals to the VME Trigger Modules (FVME TMWR) in each crate of specific subsystem and receive XOFF signals. FVME TMWR module distributes Trigger and Spill signals to all DRE modules within one VME crate over TTC bus and receives XOFF signals form these modules over VME 64x bus.

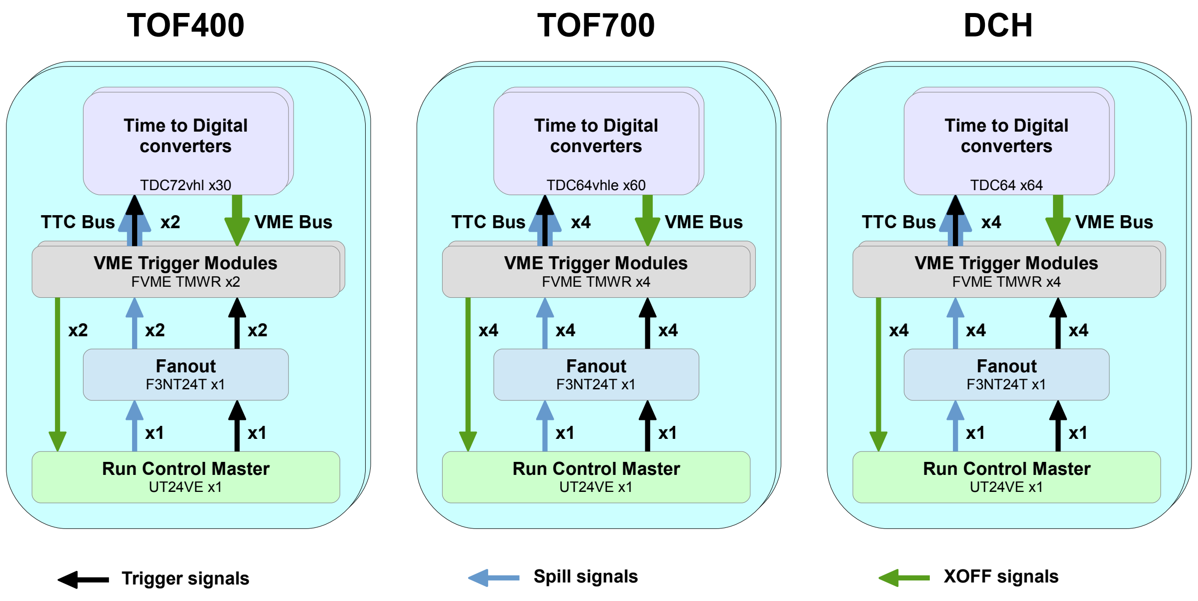


Figure 41: BM@N TDC-based Trigger Distribution

18.4.2 ADC-based subsystems

There are 5 ADC based subsystems at BM@N: GEM, CSC, STS and ECal, FHCal.

DRE modules are installed into VME crates at the first three subsystems. VME crates provide cooling and power supplying for them. Trigger distribution architecture is similar. The Run Control Master (LTU) module with (without at STS) fanout distribute Trigger and Spill signals to the Run Control Slaves modules and receive XOFF signals from them. Run

Control Slaves modules with fanout modules, in its turn, distribute trigger signals to ADC and Front-End Control modules and collect XOFF signals form them.

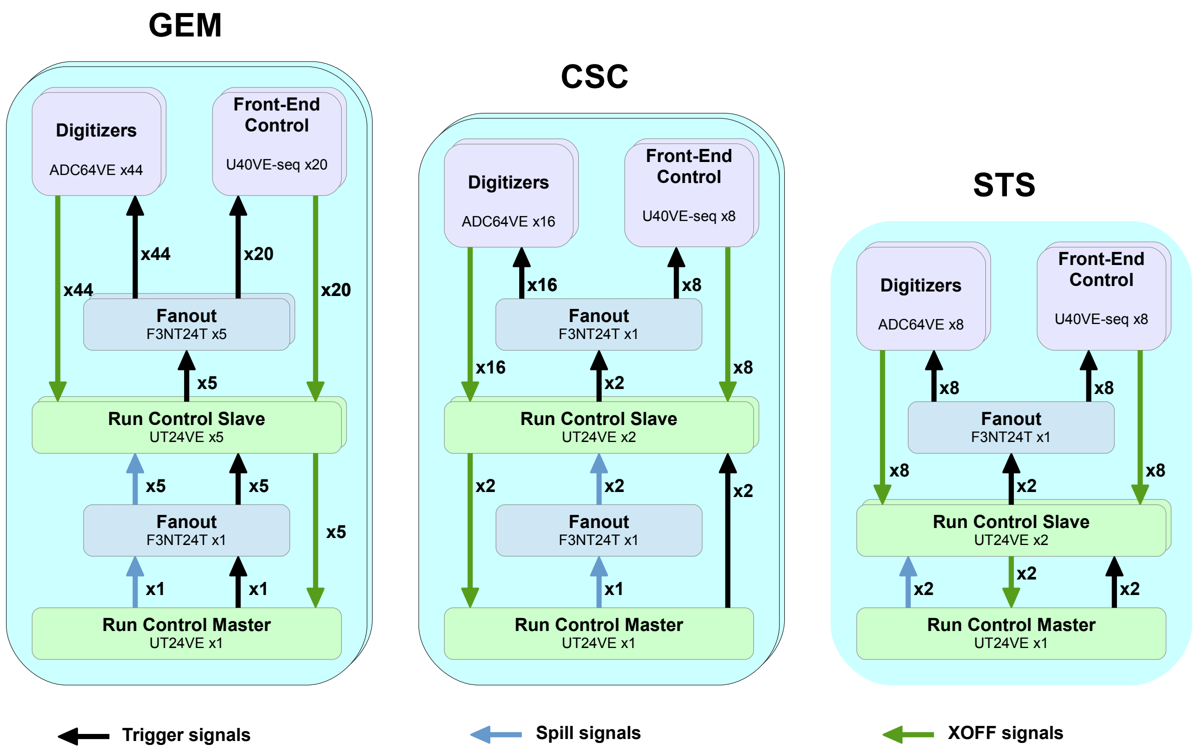


Figure 42: BM@N ADC-based Trigger Distribution

ADC modules at ECal and FHCal subsystems are placed near detectors and Trigger Distribution modules are installed into VME crates for cooling and power supplying. The Run Control Master module distribute trigger signals to The Run Control Slave modules and receive XOFF signals from them. The Run Control Slave modules and fanout transmit trigger signals to ADC module over 20 meters coaxia l cables and collect XOFF signals form them.

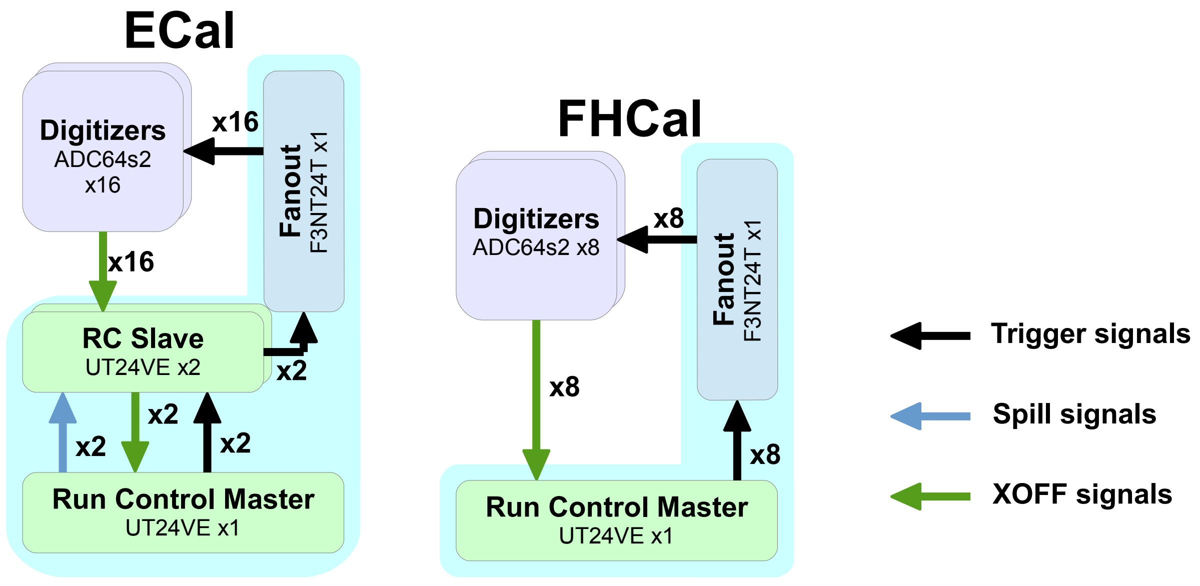


Figure 43: ECal and FHCal Trigger Distribution

18.4.3 BM@N MWPC detector DAQ

MWPC DAQ subsystem includes HRB6ASD readout modules, TTB9V synchronization modules and local VME subsystem. Raw data are pushed down via 1000Base-T Ethernet using UDP/IP and M-Stream protocols. TTB9V modules are used for trigger and timing distribution to HRB6ASD readout modules. Also TTB9V collects busy signals from readout modules. All TTB9V modules are instaled into local VMEDAQ system that used for connecting to trigger and timing distribution networks of BM@N experiment.

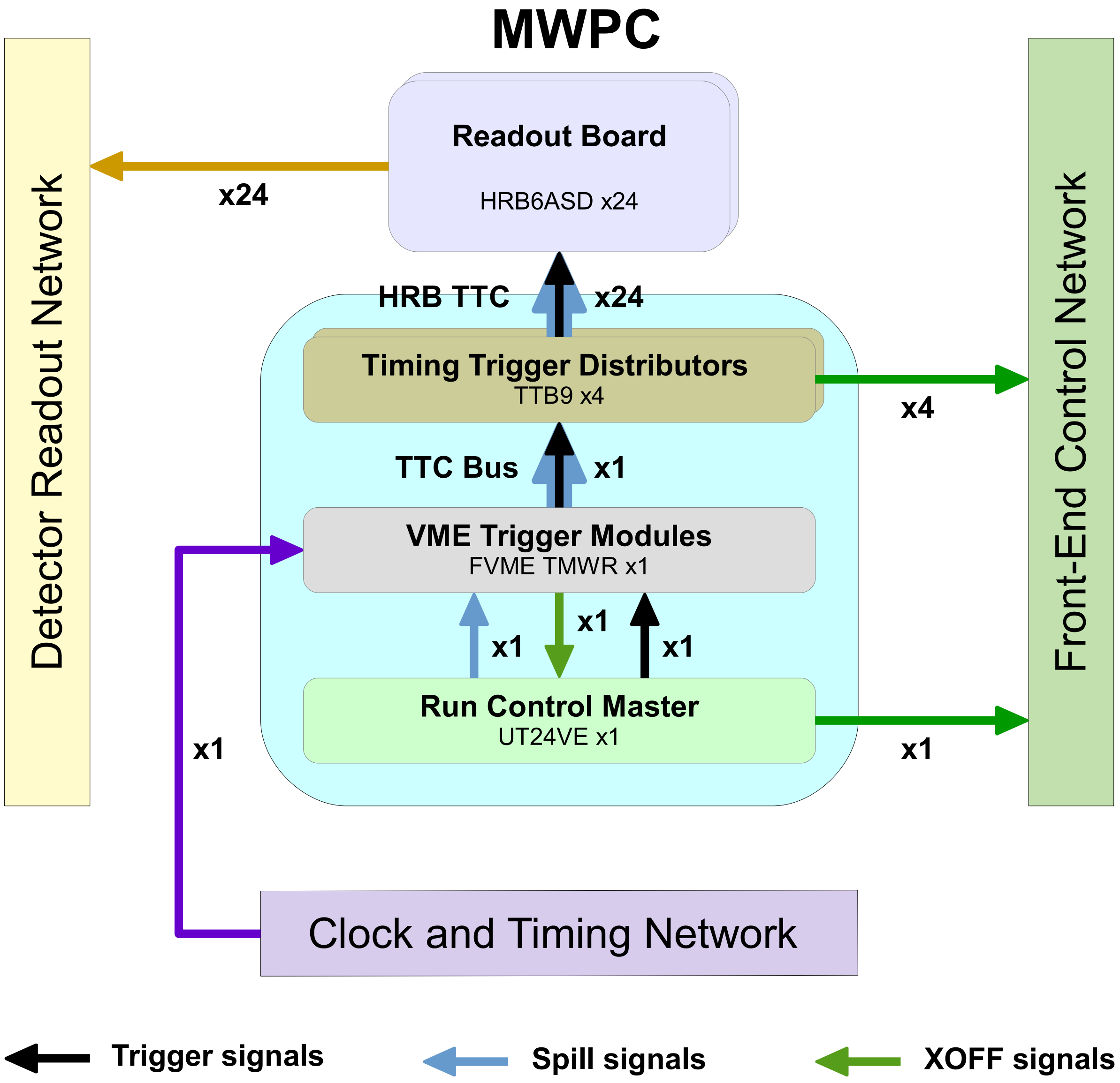


Figure 44: MWPC Trigger Distribution

19 References

[1] MPD collaboration, "MPD Conceptual Design Report v1.4," tech. rep., JINR, Dubna, 2013.

[2] M. Mota *et al.*, "A flexible multi-channel high-resolution Time-to-Digital Converter ASIC," in *Nuclear Science Symposium Conference Record, IEEE*, pp. 9/155 – 9/159 vol.2, 2000.

[3] I. Slepnev, "HWIP Wiki." `https://afi-project.jinr.ru/projects/hwip/wiki`, 2015. [Online; accessed 29-May-2015].

[4] J. Serrano, P. Alvarez, M. Cattin, E. G. Cota, P. M. J. H. Lewis, T. Włostowski, *et al.*, "The White Rabbit Project," in *Proc. of ICALEPCS TUC004, Kobe, Japan*, 2009.

[5] OHWR, "White Rabbit Node Core Wiki." `http://www.ohwr.org/projects/wr-switch-hw/wiki`, 2015. [Online; accessed 29-May-2015].

[6] T. Włostowski, J. Serrano, G. Daniluk, M. Lipiński, and F. Vaga, "Trigger and RF Distribution Using White Rabbit," in *Proceedings of ICALEPCS2015, Melbourne, Australia - Pre-Press Release*, 2015.

[7] Community, "Ceph." `http://ceph.com/`, 2015. [Online; accessed 29-May-2015].

[8] A. S. Vestbø, "Pattern Recognition and Data Compression for the ALICE High Level Trigger," 2004.

[9] E. Strohmaier *et al.*, "TOP500 website." `http://www.top500.org`, 2015. [Online; accessed 26-Dec-2015].

[10] Z. LLC, "Zabbix." `http://zabbix.com/`, 2015. [Online; accessed 29-May-2015].