The DAQ System for a Beam Detection System Based on TPC-THGEM

Si Ma, Fei Li, Wei Shen, Zhe Ning, Yuguang Xie, Kejun Zhu

***Abstract*—A beam detection system has been developed for tracks, position and energy calibration of electron, which consists of a TPC detector[1], a PDD[2], a detector to be calibrated and a silicon-based energy detector.**

Its data acquisition(DAQ) system aims to realize readout, event built, online monitoring and reconstructing for 511 channels of TPC and 200 channels of PDD in 100Hz trigger rate provided by the electron beam device.

In our design, the DAQ system is a small scale distributed system, which has been divided into two parts. One part communicates with readout electronics, based on client-server architecture, we call it ROS(Readout System). The other part receives sub-event from ROSs, merges and processes event data in each DAQ component based on DAQ-Middleware software framework, we named it ODP(Online Data Processing). The two parts running independently, which improves the performance, scalability and maintainability of the whole system.

This paper will present the system and software architecture design, implement of the DAQ software and evaluation of its ROS performance.

***Index Terms*—Data acquisition system, TPC, Beam detection**

I. Introduction

THE beam detection system is aimed to study the effect of location, energy and direction factors of electron beam on the detector. It consists of a electron beam equipment and four detectors. The beam device provided low-energy electron beam ranged from 0.1Mev to 3Mev and high-energy electron beam ranged from 3Mev to 50Mev. Four detectors are a TPC detector, a PDD detector, a detector to be calibrated and a silicon-based energy detector. The detectors can be deployment into several different combinations as required.

The TPC detector is the main detector, it aims to measure tracks of electron online. The PDD detector is a position detector, played the following effect in this system.

i To determine the difference between extended position and reconstructed position when low-energy electron passing through the TPC, as well as the probability of electron scattering after piercing the exit window of the TPC.

ii To determine the accuracy of the scanning magnet positioned on high-energy electron, then to determine whether we can remove the online TPC detector for high-energy position calibration of electron.

Manuscript received May 30, 2016.

The authors are with the e Institute of High Energy Physics, Chinese Academy of Sciences, P.O.Box 918-1 100049 Beijing, China (telephone: +86-10-88236422, e-mail: masi@ihep.ac.cn).

[1]A thick gas electron multiplier detector.

[2]A particle distribution detector.

To achieve the spatial resolution requirements, as well as reducing the number of channels of electronics, the TPC was designed with several pads, with which measure tracks of electron in sample. The structures and principles of TPC and PDD are shown in Fig.1.

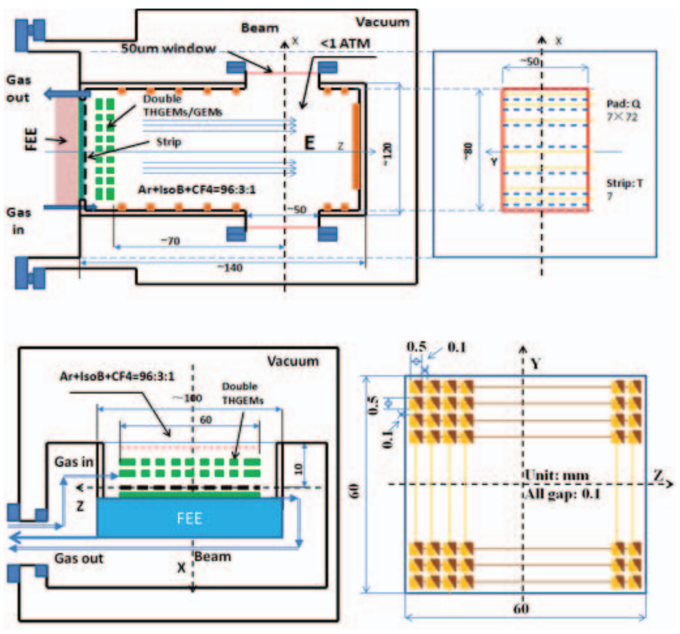


Fig. 1. Structures and principles of TPC and PDD

DAQ system is the interactive interface with front-end electronics and detectors. It is used to read data from the front-end electronics modules resided in the VME crates, concatenate the data fragments into a subsystem event in parallel, then transmit to the back-end system to do data stream merging, monitoring and recording.

II. System Requirements

The experiment requires ROS of each detector running independently, configure the parameters of detectors and front-end electronics conveniently and system control should be flexible. Detailed introduction of the system requirements will be presented from the following aspects.

A. Run Mode Requirements

The DAQ system should be capable of taking data from detectors with several different combination modes. TABLE1 shows the summary of run mode requirements.

TABLE I
SUMMARY OF RUN MODE REQUIREMENTS

Run Mode	TPC	PDD	Be Calibrated	SiD
Online Calibration Mode	Y		Y	
Independent Calibration Mode			Y	
Debugging Mode I	Y	Y		
Debugging Mode II	Y			
Debugging Mode III		Y		
Energy Calibration Mode				Y

B. Readout System Requirements

There are many front-end electronics modules for TPC and PDD detectors. Each module consists with four sub-plates, and each sub-plate which has its own IP address can creates a data packet with eight channels when trigger signal arrives. It processes the packet by means of an FPGA (User-FPGA) and transfers data to a computer via ethernet with the SiTCP protocol[1], which is a hardware implementation of TCP/IP on a FPGA device, enabling it to achieve high speed data transfers.

ROS should be able to readout 511 channels of TPC and 200 channels of PDD electronics with 100Hz trigger rate. That is, ROS need to finish data receiving from all of electronics modules, building them to a complete sub event and integrating sample values of each channel of 100 events in at least one second. Finally, the event data should be sent to ODF softwares.

C. Control GUI Requirements

1 Run number, the number of events, trigger rate and detection efficiency should be all displayed on the screen in real time. Q and T values spectrograms and hit maps of all channels are required.
2 Reconstructed tracks and hit location of electron should be displayed once per second.

D. Other Requirements

Each raw event data needs to be reconstructed in order to find good events. Parameters such as the average drift velocity, track projection distance, operating voltage etc. should be configured by users conveniently.

III. SYSTEM ARCHITECTURE DESIGN

The DAQ system architecture is designed to a multi-level system using advanced commercial computer and network technology as shown in Fig.2.

According to the site locations, DAQ system can be separated into two parts: VME front-end system and back-end system.

In front-end system, all electronics modules of TPC and PDD are inserted into two VME crates. VME crates only provide power for the modules, the computer used to read data communicates with electronics modules via a cisco switch. We also provide a independent control terminal for each detector to facilitate independent control and debugging.

In back-end system, we used two IBM System x3750 M4 servers and two industrial control computers. One of the

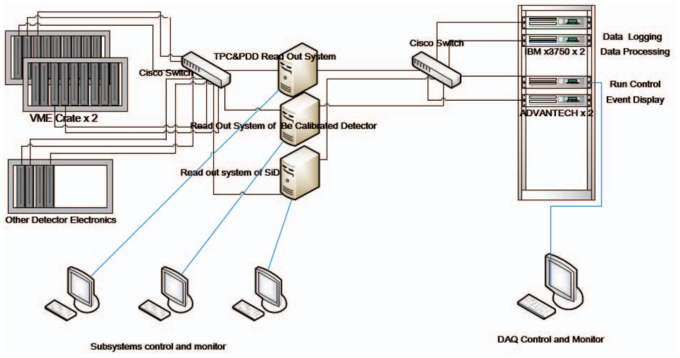


Fig. 2. DAQ system architecture design

servers is used as event builder component and logger component while the other one is used for online reconstructing component. One of the control computers is used as a DAQ operator, which can control all DAQ components by sending messages, while the other computer can be used for display.

All of computers used in this DAQ system are running Linux operating system.

IV. SOFTWARE ARCHITECTURE DESIGN

The data flow component diagram of this DAQ software is shown in Fig.3, which is based on DAQ-Middleware software framework[2]. DAQ-Middleware is network-distributed DAQ system, which implements with C++ and Python programming languages and CORBA for its communication protocol[3].

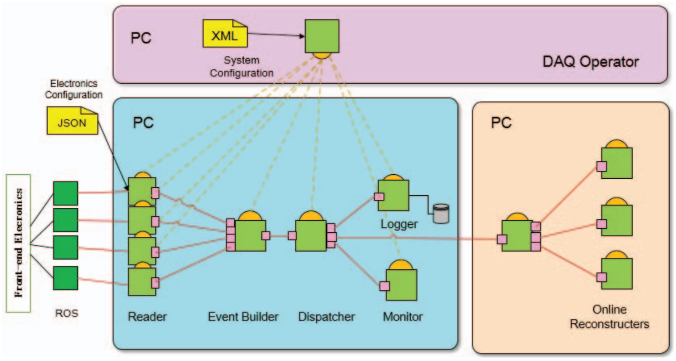


Fig. 3. DAQ software architecture design

Readout systems read data from several front-end modules, build to event and send to back-end. Reader components receive sub-events then send them to event builder. Event builder component merges sub-events together to obtain complete events and sends them to a dispatcher, which can distribute event flow. Monitor component used for online Monitoring with the conditions of electronics and detectors.

The ROS is designed based on common client-server architecture, data transmitting between client and server using usual sockets, its software implementation is shown as Fig.4.

Because of too many channels of a detector, and one electronics sub-plate can only readout data of eight channels, so each ROS needs to establish too many socket connections with the electronics modules. In order to reduce the number of

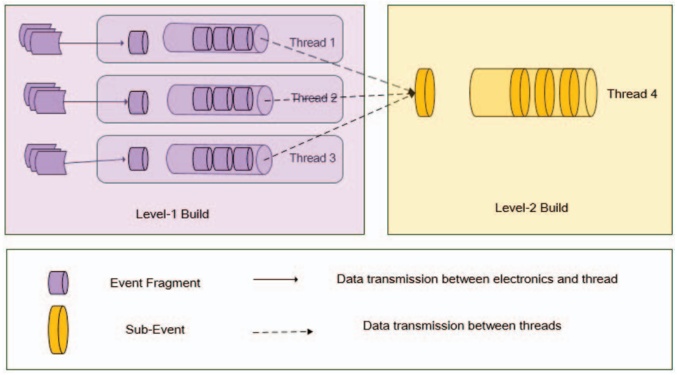

Thread 1
Thread 2
Thread 3
Thread 4
Level-1 Build
Level-2 Build
Event Fragment
Data transmission between electronics and thread
Sub-Event
Data transmission between threads

Fig. 4. ROS software implementation

receiving thread, we designed with two level event building. Data shared between threads by queues.

For level-1 building, in which data packed within receiving threads, data packets were received from electronics sub-plates in order, and built to a event fragment by the same event number, then finally inserted it into a queue. The number of the socket connections can be flexibly configured for each thread in order to achieve optimal performance.

For level-2 building, in which data packed between several receiving threads and a packing thread, each event fragment with the same event number will be popped from queues mentioned previously, then packed together to create a complete event for a detector and finally inserted it into queues for logging and sending.

For ODP, there are several DAQ components controlled by a DAQ operator component, which sending commands to all other components. In the DAQ-Middleware architecture, each DAQ component designed with four state transition, they are loaded, configured, running and paused in DAQ life cycle. State transition and method called diagram is shown in Fig.5.

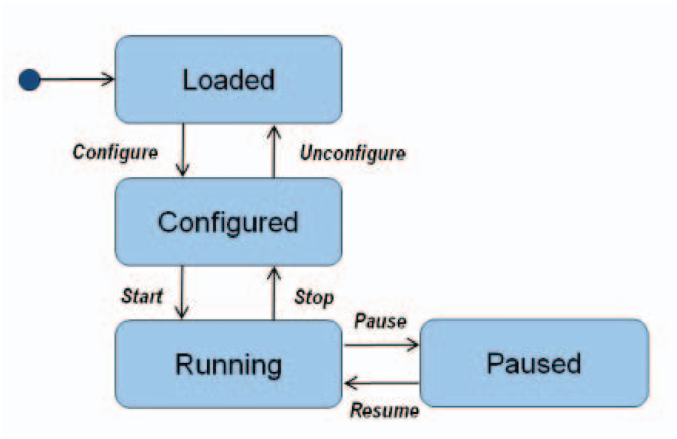
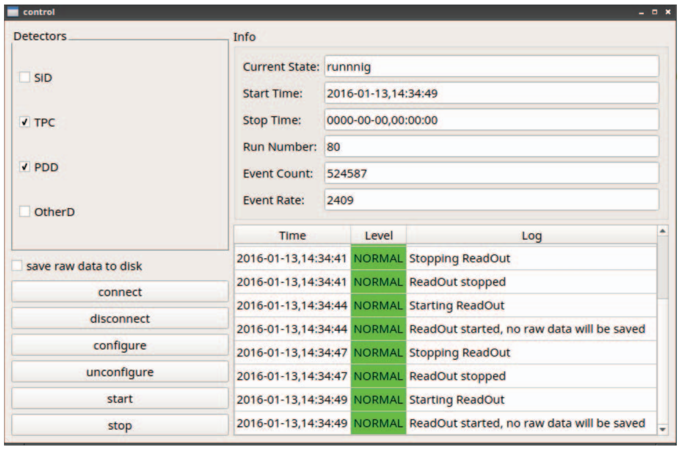


Fig. 5. DAQ state transmission

We designed to use usual socket to transmit data between components because of the improvement performance research in[4].

The control GUI of this DAQ system is shown in Fig.6, which communicated with DAQ operator is designed with Qt5.5 running on Linux operation system.



Current State: runnnig
Start Time: 2016-01-13,14:34:49
Stop Time: 0000-00-00,00:00:00
Run Number: 80
Event Count: 524587
Event Rate: 2409

Time	Level	Log
2016-01-13,14:34:41	NORMAL	Stopping ReadOut
2016-01-13,14:34:41	NORMAL	ReadOut stopped
2016-01-13,14:34:44	NORMAL	Starting ReadOut
2016-01-13,14:34:44	NORMAL	ReadOut started, no raw data will be saved
2016-01-13,14:34:47	NORMAL	Stopping ReadOut
2016-01-13,14:34:47	NORMAL	ReadOut stopped
2016-01-13,14:34:49	NORMAL	Starting ReadOut
2016-01-13,14:34:49	NORMAL	ReadOut started, no raw data will be saved

Fig. 6. Control GUI

V. Performance Evaluation of Readout System

A performance measurement was performed to evaluate the throughput of readout system. In the test, two PCs connected with each other through a 10Gb switch which is shown in Fig.7.

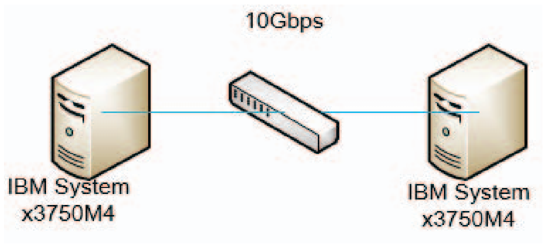


Fig. 7. A measurement setup

For measurement, the sender is a simulation one, which sending packets in parallel as fast as possible with multi threads, as well as the receiver receiving, building packets to complete events without recording.

The performance measurement was performed with using usual socket transmitting based on client-server architecture on CentOS 7 Linux . The specifications of sender and receiver are shown in TABLE2.

TABLE II
Specifications of two computers in measurement

OS	CentOS Linux 7 64bit
CPU	Intel Xeon CPU E5-4610 v2 2.13 GHz
Memory	64GByte
Network	10Gbps

The throughput of readout system without recording is measured, and results is shown in Fig.8. The average event rate in the measurement is shown in Fig.9.

As shown in the figures, the throughput of ROS is closed to the limit of the 10Gb network.

VI. Summary

Based on DAQ-Middleware framework, DAQ software of beam detection system achieved. Readout system and each

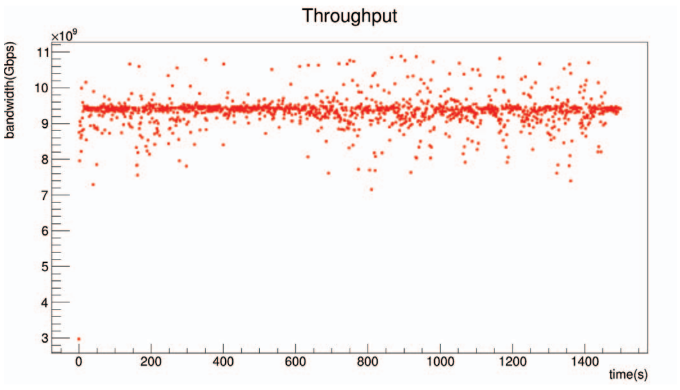


Fig. 8. A performance measurement result I

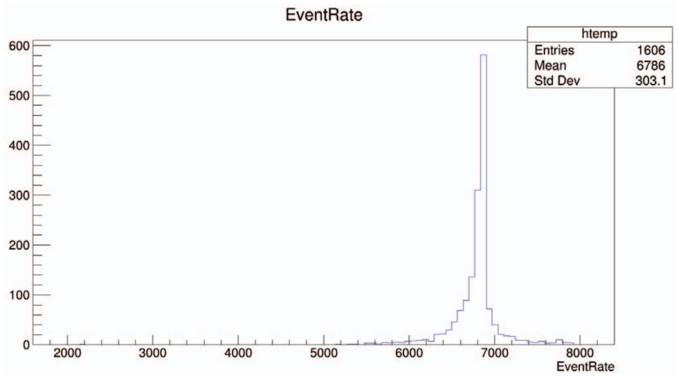


Fig. 9. A performance measurement result II

component of back-end data flow could be controlled and monitored independently. The test result shows that the DAQ performances well and the design can meet current experiment requirements.

References

[1] T. Uchida, Hardware-Based TCP Processor for Gigabit Ethernet, IEEE Trans. Nucl. Sci. NS 55 (2008) 1631.

[2] Y. Yasu, K. Nakayoshi, E. Inoue, H. Sendai, H. Fujii, N. Ando, T. Kotoku, et al., A Data Acquisition Middleware, in Proc. IEEE/NPSS Real Time Conference, pp. 1-3, May 2007.

[3] Y. Nagasaka, H. Maeda, H. Sendai, E. Inoue, T. Kotoku, N. Ando, S. Ajimura, M. Wada, Communication architecture of DAQ-Middleware, Proc. IEEE/NPSS Real Time Conference, pp. 1-3, 2012.

[4] Y. Nagasaka, H. Maeda, H. Hori, H. Sendai, E. Inoue, E. Hamada, T. Kotoku, S. Ajimura and M. Wada, Performance Improvements of DAQ-Middleware, IEEE, 2014.