New data acquisition system for CERES/NA45 at CERN

P. Braun-Munzinger, J. Holeczek, D. Miśkowiec, H. Sako, GSI
H. Tilsner, H. Wessels, University of Heidelberg

For the 2000 run of CERES a new TPC readout scheme was implemented [1]. The data acquisition system (DAQ) needed to be adapted to the new readout. Between March and September 2000 we designed and implemented a completely new DAQ. It was then successfully used to collect 30 millions central Pb+Au events in the 160 GeV/nucleon run. Below we briefly describe the DAQ scheme and performance. More details can be found in [2].

The new DAQ was to a large extent based on standard hardware components (20 personal computers under linux) and software tools. It was highly modular so we could optimize one part of the system without touching the others. Ten readout PCs were collecting events during the 5 s long SPS burst. In the 14 s long burst pause they were sending the collected data via ethernet to an event builder PC in the CERN Central Data Recording facility (CDR). There the ten data buffers were merged into one and saved on disk. The tape daemon, asynchronously running on this machine, copied the file to tape. The cleaner process was deleting the oldest files such that half of the disk space was always available. Below we go through the collecting, storing, sending, and event building stages in more detail.

Charges induced on the TPC readout pads were digitized in FEDC modules [3] sitting in VME crates in the experimental zone. Each VME crate was connected via a MXI interface and a 20 m cable to a readout PC, located in the counting room. Eight readout PCs were needed to read the 16 TPC chambers. Other CERES detectors, SDD, RICH1, and RICH2, sent their data via receivers to memory modules, then via optical links and O2PCI modules to the memory of an embedded PC. The beam detectors ADCs and TDCs in three daisy-chained VME crates were accessed by yet another readout PC via PVIC.

The readout was triggered by an external signal applied to an input channel of an I/O card (PC36C by Eagle Technology) plugged in each readout PC. The collector software was polling on the bit. Once a trigger has been seen, the PC would set a busy signal on an output line of the I/O card. A logic OR of all busy signals went to the trigger system and inhibited new triggers. The busy signal was removed only after the complete data had been in the memory of the PC (unless in the pipeline mode, see below).

The event size was 0.5 MB. The average busy duration, i.e. the average time needed to get the event into the memory of a readout PC, was 1.7 ms for the beam detector crates, 3.8 ms for SDD, 2.3 ms for RICH, and 5.7 ms for TPC. The largest fraction of the latter was spent in the data transfer from the FEDC to the readout PC via MXI (12 MB/s). With the beam intensity of 10^6 per burst, and with the centrality trigger of 8%, the rates of offered and accepted triggers per burst were 1000 and 300, respectively. In the middle of the run a pipeline readout was implemented for the TPC. The 5.7 ms were split in two parts: $\sim$1 ms, needed for the data to get in the ALTRO chips of the FEDC, and $\sim$5 ms for the transfer to the readout PC. The number of accepted triggers increased to 400 per burst.

On all readout PCs the data were collected in the upper most 64 MB of the total 128 MB physical memory. This memory was disabled for linux by an appropriate entry in `lilo.conf`. A memory device driver, similar to `mem.c`, was used to access this area via `/dev/daqmem`. From the system level the user could handle `/dev/daqmem` like an ordinary disk file, including dumping, editing, copying, etc.

The collecting was controlled by a set of parameters, residing in the kernel memory. After a trigger, the collector process would read them to know where to store the event. After storing it would update the appropriate numbers. Later, when the burst was finished and the data needed to be sent to CDR, the sending routine would read the parameters to figure out how many bytes to send. The access to the parameters was provided via a simple device driver. From the system level the user could access the parameters at any time via `/proc/daqctrl`.

In the burst pause the readout PCs sent the collected data to one event builder PC in CDR via sockets. The 10 receiver processes, running on the event builder machine, dumped the data into named pipes (FIFOs). The sending speed of each readout PC was limited to about 10 MB/s (fast ethernet). The Gigabit ethernet card on the CDR side, however, was limiting the total transfer rate to 30 MB/s. Furthermore, because of the run control overhead and because of activities on the event builder PCs, only 200-250 MB of burst data could be sent within the burst pause in a stable mode. This was the bottle neck of the DAQ. The resulting continuous data rate was 10 MB/s.

The event builder (evb) read the FIFOs, checked the consistency of event counters, merged the subevents, and saved the output, containing one complete burst, to a disk file. The processing speed was 15-17 MB/s and thus the data of the next burst had to be sent to a different CDR machine. In total seven event builder machines were used.

A tape control script sorted link files created by evb into tape job queues and submitted them on each evb machine. Typically there were 1-3 jobs per machine. The actual copying to tape was done using CERN `tpwrite` with 5 MB/s. The overal taping rate was limited to 20 MB/s.

The new DAQ, in spite of various problems, was working from the beginning of the heavy ion run and allowed to collect total of 30 millions events.

References

[1] H. Tilsner, GSI Annual report 2000.
[2] http://www.gsi.de/ misko/ceres/daq/note.html
[3] Design by Luciano Musa, CERN.