Hall D Event Unblocking

Elliott Wolin and David Lawrence

10-May-2012

GlueX Doc #1886 covering 12GEV line 1532030

Below I define what is meant by event "unblocking", argue why it is useful, discuss where it might happen in the event chain, and describe our plans for event unblocking.

Why Event Data Becomes "Blocked" or "Entangled"

For efficiency event data in the front-end modules will be read out in "blocked" mode. Here the first module in the crate will transmit all data for N events in a single block, the second module will do the same, down the line until all modules have delivered data for N events to the ReadOut Controller (ROC). In the simplest case the ROC will pack all data from all modules sequentially into a single buffer, then transmit the buffer to the event building system. The event building system will then collect data from all ROC's for the N events into a single buffer.

At this point data from N events is mixed or "entangled" in a large buffer, and data from a single event appears in many discontinuous pieces in the buffer. Processing of single events by processes downstream of the event builder system thus can become complicated. Disentangled events would be contiguous in memory and in sequential order, and would be much simpler to process.

Why Unblocking is Useful

At low luminosity a fraction of all events must be monitored in the monitoring farm, and at high luminosity all events must be analyzed in the L3 farm. Note that about 90% of the events will be rejected by the L3 farm. If a monitoring or L3 farm node receives events in entangled blocks it will have to disentangle them prior to processing. Disentangling is conceptually straightforward but tricky due to the nature of the front-end boards and readout libraries.

Further, it is much simpler to extract single events from the data stream to send to special monitoring or calibration processes, alternate output streams, etc. if the blocks are disentangled. Finally, if event blocks are not disentangled when they are written to disk then the offline analysis routines would have to repeat the tricky disentangling procedure.

Where Might Unblocking Be Performed

Unblocking can be performed in a number of places:

1. front-end ROC
2. event building system
3. process immediately downstream of the event builders
4. monitoring/L3 farm
5. event recorder
6. offline analysis programs

Unblocking in the ROC (1) would require adequate CPU resources beyond what is needed to read out the front-end modules and ship data over the network. Dual-core ROC's do not have enough, but it is likely the recently available quad-core ROC's have enough cpu power available. The JLab DAQ group is currently testing quad-core processors and will determine if unblocking on quad-core ROC's is feasible.

Modifying the event builders to add an unblocking capability (2) is less desirable than adding an additional processing step after the event builders (3) running on similar high-power multi-core processors, as there would be no need to make custom modifications to the standard event builder.

Unblocking in the monitoring/L3 farm (4) is problematic for a number of reasons. It is not clear the monitoring farm will have enough cpu power nor whether all events will be processed prior to the event recorder in the early stages of the experiment. Further, the unblocking code would have to be incorporated into the standard Hall D analysis framework.

Unblocking in the event recorder (5) or in offline analysis programs (6) is redundant because events must be unblocked at an earlier stage to allow for monitoring and L3 analysis.

Current Plans for Event Unblocking

The two most attractive places to perform unblocking are in the front-end ROC (1) or in a special process just downstream of the standard event builders (3). The former would happen

in a custom secondary readout list, the latter in a custom EMU (i.e. the framework used to write event builders and event recorders).

The JLab DAQ group is designing and prototyping unblocking code for Hall D in such a way that it could be run in either place. They further will evaluate CPU resources needed for both. Current plans are to have the DAQ group write the unblocking code for Hall D when we determine which place is most suitable. Note that DAQ group personnel are the most qualified to do this since they have developed both the front-end readout software and the event builder system.

Recently CLAS performed unblocking in the ROC stage in a production system and found it took about 10% of the cpu resources to complete. Although the CLAS environment was somewhat different from ours, we believe it will not take more than double that of CLAS. Even the doubled cpu load is acceptable for Hall D, especially since our cpu's will likely be substantially faster than the ones CLAS used. Thus our tentative conclusion is that unblocking will be performed in the ROC.

Manpower Estimates

The 12GeV schedule allocates 9 man-weeks for planning and 12 man-weeks for writing the unblocking system, plus some fraction of 11 man-weeks allocated for DAQ checkout. This appears adequate at this point.

The JLab DAQ group will perform the initial planning for event unblocking, and we will rely on them to evaluate the suitability of performing unblocking in the ROC or in an EMU stage. We further plan to have the DAQ group write the unblocking code once we determine where it should reside. If the DAQ group is unwilling or unable to do this using Hall D personnel will do this instead.

Finally, the work reported here completes the event blocking planning stage.