Diagnostic and Calibration Tools for the Run 2 Muon System

DØnote 3062

Boris Baldin, Oliver Bardon, Sten Hansen, Darien Wood

August 24, 1996

1 Introduction

This is a preliminary survey of the sort of tools which might exist in Run 2 for diagnostics and calibration of the muon system detectors and electronics. Any comments and suggestions are welcomed. This document will be updated as soon as new considerations are taken into account. Several of the components involved have not yet been designed, so much of this is expected to change. This survey might be useful in anticipating some of the features which might be useful for diagnostic purposes.

2 Relevant Hardware

A general description of the readout electronics is needed in order to discuss the diagnostic and calibration features. Some of the technical details exist in DØnotes #2370, #2705, #2780, #2888, #2988 and in the preliminary specifications of scintillator electronics and Muon Readout Cards.

Block diagrams for the readout electronics for the three types of subsystems are shown in in Figures 1, 2 and 3. Only the components relevant to diagnostics and calibration are featured.

Pulsing: All front end channels (PDT, MDT, and scint) will be equipped with pulser inputs. The pulses themselves will be generated on the Front-End Units: Front End Board (FEB) for WAMUS PDTs, Scintillator Front-End Card (SFC) for scintillator, and MDT Card (MDC)

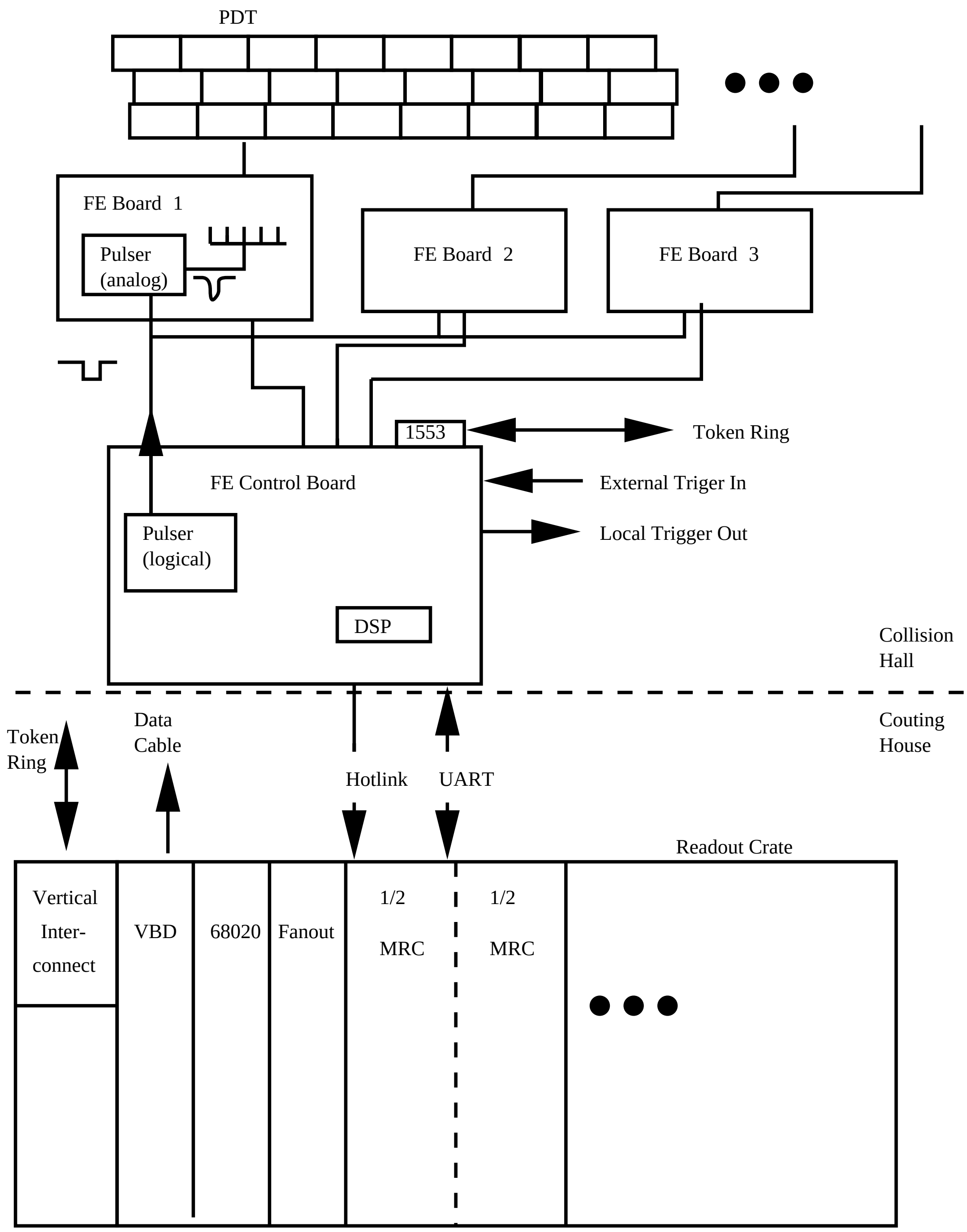

Figure 1: A block diagram of the readout of the WAMUS PDTs showing some features related to diagnostics and calibration.

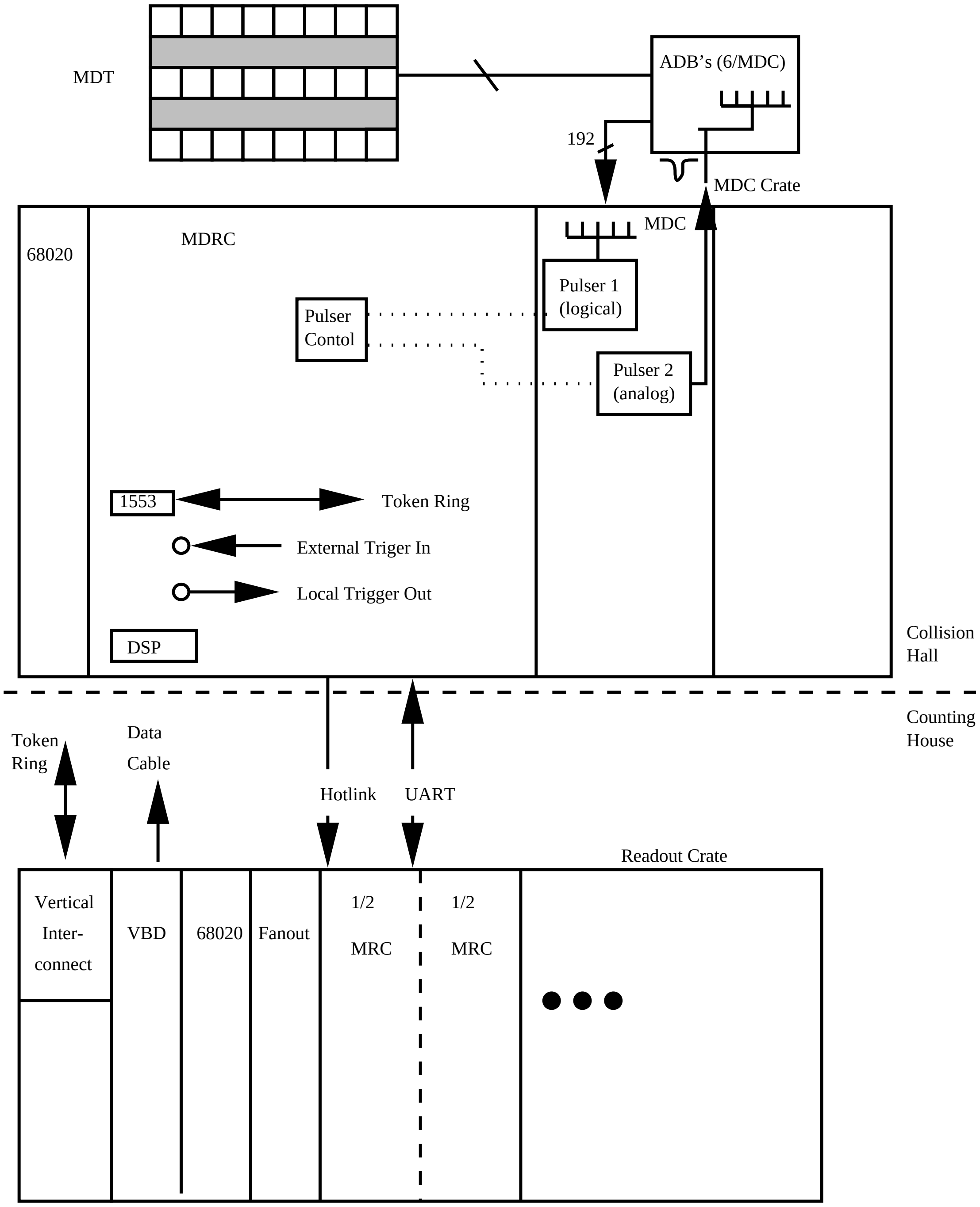

Figure 2: A block diagram of the readout of FAMUS MDTs showing some features related to diagnostics and calibration.

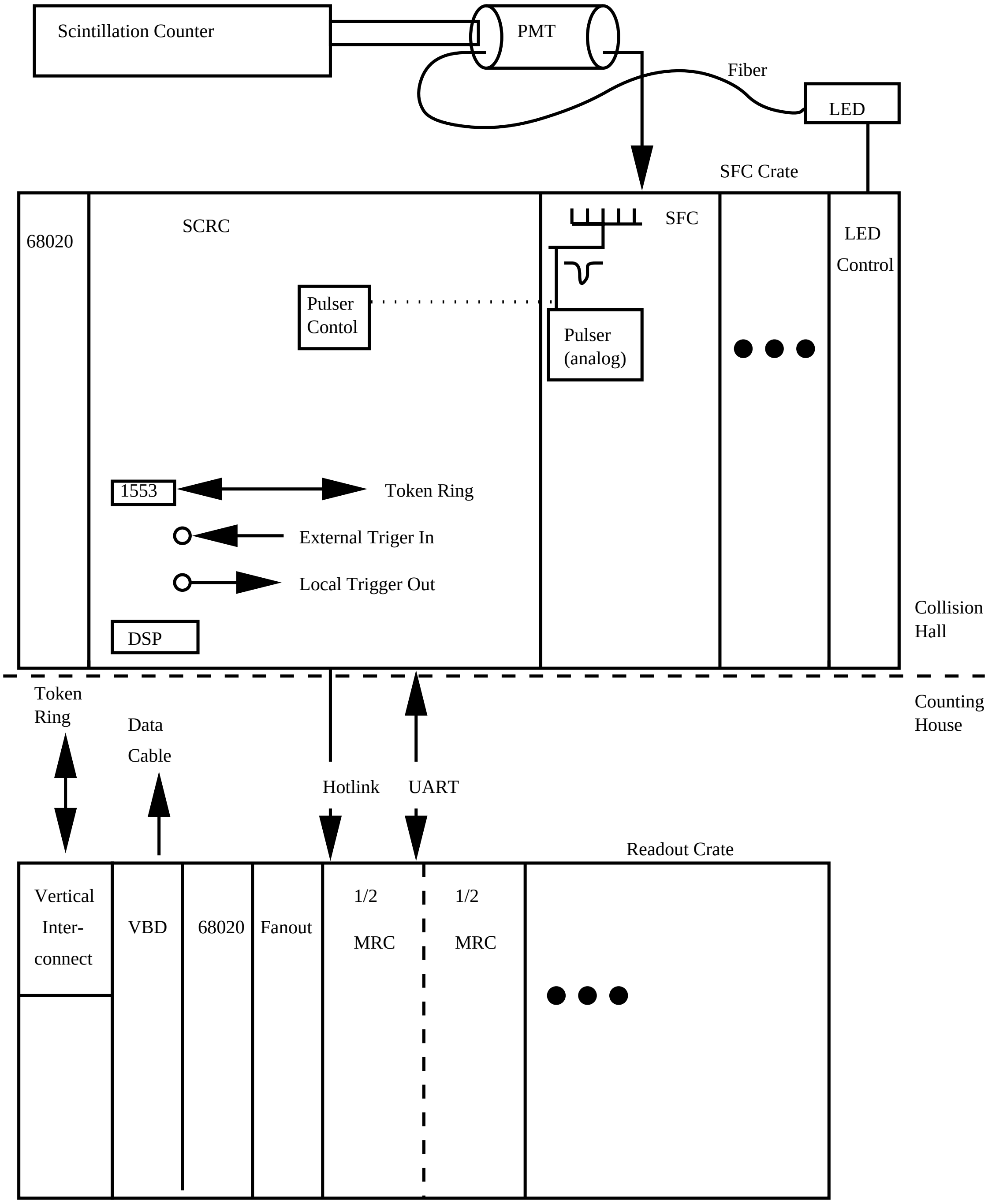

Figure 3: A block diagram of the readout of muon scintillator showing some features related to diagnostics and calibration.

for the FAMUS MDTs. The enabling, delays and amplitudes of the pulsers will be controlled from the Control Units: the Control Board (CB) for the WAMUS PDTs, Scintillator Readout Controller (SCRC) for the scintillator, and the MDT Readout Controller (MDRC) for the FAMUS MDTs. For the MDT electronics, there are actually two sets of pulsers: an analog pulser that is send to the Amplifier Discriminator Boards (ADBs) and a logical pulser that directly pulses the inputs on the MDC (See Fig. 2). The pulser signals are broadcast to all of the front end channels on a given Front End Card/Board, but any pattern of active channel outputs can be configured by downloading an appropriate mask. The delays have not yet been specified, but it is expected that the smallest adjustment will be around 2ns.

Flashing: In addition, for the scintillators, a system is planned to inject LED flashes onto the PMT photocathodes. A preliminary description of this system is contained in a separate note.

Self triggering and external triggering: All front ends will have the ability to self trigger on single channels (noise, cosmics,...). For the WAMUS PDTs, there will also be the ability to trigger on n-deck coincidences (n=0,1,2,3,4). In addition, each control unit will have an external trigger input allowing trigger from any external source.

Processing: The data from diagnostic running or calibrations can be analyzed in the front end Digital Signal Processor (DSP), in the 68k in the readout crate, in the Level 3 nodes, or on the host computer.

3 Levels of testing

We anticipate four levels at which diagnostics and calibrations might take place. We list these starting from the most local level and moving to the most global level, which is probably the order in which these tools will be developed.

Front End Level: Each Control Unit will equipped with a DSP, which has 32 Kbytes of program memory and 48 Kbytes of data memory. The primary purpose of the DSP is to buffer the data from the front end while a Level 2 decision is pending. The DSP, however, could run diagnostic sequences when it is not in data-taking mode. These diagnostics could involve the pulsers, or self-triggering on noise or cosmics. The DSP could accumulate statistics on each channel and later pass the results down through the readout system. It might also be possible to calculate and store calibration constants in the DSP. In any case, it is anticipated that the calibration constants will be available to the DSP when it is formatting and buffering data, so that it may do pedestal subtraction, etc. The DSP will be able to control the pulser lines, thus allowing event data to be processed before triggering the next event. In this way, each DSP could be performing independent diagnostics/calibration on each module at any time when the experiment is not taking data. Systems required:

- detector module (or pulser)
- Front-End Unit
- Control Unit

Read-out Crate Level: As in Run 1, each read-out crate will have a 68k processor. This processor could run programs to accumulate data from any of the MRC's in the crate, just as it did from the MAC cards in Run 1. This would be the run 2 version of "daq_local" if we have self-triggering capability in the front ends. The RS-232 port on the 68k can be connected to a terminal to provide a control interface and display results. Systems required:

- detector module (or pulser)
- Front-End Unit
- Control Unit
- MRC and Readout crate

Read-out Crate + Vertical Interconnect Level: This is the same idea as the popular "mu_local" (aka "vax_local") program of run 1. The 68k will still manage the accumulation of data as in "daq_local", but instead of analyzing the data itself, it will pass it on through the vertical interconnect and the token ring to the host computer for more sophisticated analysis and optional recording of events.

- detector module (or pulser)
- Front-End Unit
- Control Unit
- MRC and Readout crate
- Vertical Interconnect (VI)
- host computer

DØdaq level: Here, the triggers will be managed by the framework and the data will be read out in the normal way through the VBD onto the data cables. This will require the resources of essentially the whole DØ data aquisition system. On the other hand, this is the only mode where we can read out multiple crates in a coordinated way. Since the elements of a trigger tend to span multiple crates, this might be the only mode in which to do trigger tests. This is also the mode in which calibration was done in run 1. Synchronization of the various elements might not be trivial. Systems required:

- detector module (or pulser)
- Front-End Unit
- Control Unit
- MRC and Readout crate
- VBD
- level 3
- trigger framework
- DØ daq
- host computer

4 More Details on Local Triggering

Local triggering is an important tool for diagnostics and was heavily used in run 1. In local triggering mode, the front ends are running continuously, until some simple condition is satisfied in the detector, such as a latch firing in one channel. This local trigger is sent to the control unit. The control unit can generate its own level 1 and level 2 accepts and rejects, so the flow of data is controlled locally. The control unit reads the data from the front end and can then either send the data to the MRC for readout, or apply more sophisticated requirements using the DSP and only send out the events satisfying these stricter conditions. The control unit notifies the MRC that is has data to be read out via either one of the five twisted-pair lines which transmit from the FE to the MRC, or over the UART serial link (this is still to be decided). The trigger is passed from the MRC to the 68k via the VME back plane, and the 68k then generates the VME interrupt and coordinates the readout.

For the WAMUS PDTs, the possible trigger conditions will be the same as in run 1: 0, 1, 2, 3, or 4-deck coincidence. The coincidence level will be downloaded to the FE through either the 1553 or the serial link. The control unit will see only triggers satisfying the selected coincidence level.

For the FAMUS MDTs, there will be only one possible trigger condition for an MDC: the OR of the 192 channels on that card. It should also be noted that will be possible to mask off any set of channels, so there will be some recourse if the triggers are dominated by a few noisy channels. Each MDRC will see the local triggers from up to 12 MDCs, and these will feed into a programmable AND-OR network. In addition, more sophisticated trigger conditions could be implemented in the DSPs. Of course, these must be subsets of the triggers generated in the upstream electronics. Early estimates indicated that something like a multideck coincidence algorithm could be executed in the DSP in a time around 10–50 μs.

The scintillator will operate similarly to the MDTs. The SFCs will generate triggers corresponding to an OR of the 48 PMT latches. These will feed into an AND-OR network on the SCRCs. Since there is only one deck of scintillation counters in a given layer, there is no obvious desire for implementing coincidence logic. The same possibilities will exist, however, and for the MDTs: channel masking, multi-card coincidences, and DSP programmed conditions.

As mentioned earlier, each control unit will have an external trigger input. This could be used, for example, to feed in the trigger from a cosmic ray telescope for chamber testing in the assembly hall. The external trigger would be the equivalent of the n-deck coincidence signal from the front ends in the case of the WAMUS PDTs, and the equivalent of the output of the control unit trigger AND-OR network in the case of the other detectors.

In addition, each control unit will have a trigger output, with the same equivalence relationship. This allows for such possibilities as using a local trigger from the MDTs to read out a nearby group of scintillators, or vice versa.

5 Rate Limitations

In general, the readout rate limitations for each level of testing will depend on the size of the events and the complexity of the algorithms used in the diagnostics. Some rules of thumb can be given for approximating the maximum event rate.

For the DSP level readout, the time spent can be estimated as follows:

1. 2 cycles/word for shift from input FIFO to buffer
2. 2 cycles/word for shift from buffer to output FIFO
3. Overhead per data set = 10-50 cycles
4. 2 cycles/word for any additional operation on each data word (e.g. pedestal subtraction

Each cycle takes 30 ns. The absolute minimum is the sum of 1, 2, and 3 above; 4 depends on the specific task. In summary,

$$\mathrm{time/trig} = (4 \times \mathrm{N_{word}} + 50 + (\mathrm{N_{word}} \times \mathrm{N_{additional\_op}} \times 2)) \times 30\mathrm{ns}$$

The timing for any specific algorithm can be determined with the DSP simulator.

Transfer from the DSP to the MRC can go at 16Mbytes/sec (125 ns/word), so this rate limit is very similar to that imposed by the DSP. Consequently, the maximum rates should be similar for the first two modes of diagnostic readout.

Transfer of events out of the readout crate is considerably slower. The Vertical Interconnect, Token Ring, and Gateway together limit the transfer rate to about 5μs/word. The rate can be further constrained if the Gateway is shared with other processes. In Run 1, we typically ran in this mode at 10's of Hertz without any problem for small local cosmic events.

In local triggering on cosmic rays, the events tend to be very small (3 or 4 hits/event) and the event rate is usually limited by the actual cosmic flux rather than any bottlenecks in the readout. Pulsing mode is the other extreme, where the occupancy can be 100%. The fastest possible readout to the MRCs would be about 10kHz for pulsing a single module, or 1kHz for pulsing a crate of MDTs of scintillators. Such events could be transferred out through the V.I. at a maximum rate of about 200 Hz for a single module. Faster readout is possible through the VBD to the data cable, but more system resources are required for this mode.

In diagnostic and calibration applications, events are usually analyzed in real time and are not written to disk. The rate limitations above do not take into account the time required for analysis done on the events in the DSP, 68k, L3 node, or host computer; only such things as buffering and formatting are included in these baseline numbers. In practice, the analysis processing time is most likely to be the rate limiting step for calibration pulsing.

6 Software Development

So far, little work has been software for run 2 diagnostics and calibration. Some of the code from run 1 might be salvageable, but much new code has to be written:

DSP: The DSP is a new item, so no code exists from run 1. The standard readout mode code is in development now. The code for diagnostic modes has not been started.

68k: Since the 68k's will be reading out MRCs instead of ADCs and MACs, that part of the code has to be rewritten. Some of the daq_local histogramming and user interface code might be salvageable. The server(client?) for the VI might be reused as well. Of course, the data format will change, and the code must reflect that. We expect that the time-critical parts of the code will be written in 68k assembly language. Compilers also exist for C and C++, and these might be more appropriate for writing the user interfaces, etc.

host computer: It is believed that the CDAQ protocol will still be used for communicating between the host computer and the token rings.

This would mean that the host-side interface of mu_local might have to undergo little change. Hopefully, many of the nice features of mu_local can be salvaged (histogramming, event displays, etc.) although the old graphics package (di3000), user interface (compac) and host operating system (VMS) will probably all have to be changed. In addition, the mu_local code will have to be extended to give appropriate displays and histograms for the new detectors which did not exist in run 1.

The program comm_tkr will be maintained and can be used to download commands to the 68k's and settings to the 1553's on the control units. This is an experiment-wide utility provided by the online software group. The hardware database is currently based on a VMS-specific implementation of RDB. This will have to be replaced. In the meantime, some means of using the physical hardware addresses for downloading is hoped to be available.

The data acquisition framework for full-system calibration is also an experiment-wide utility; we expect the online group again to provide the "calib" framework in run 2.

Exactly what the host computer will be is still uncertain, but Windows and Unix are strong candidates for the operating system.