Fermi National Accelerator Laboratory

FERMILAB-Conf-91/321

The Design of a Hardware Testing System for the D Zero Detector

R. Angstadt, M. Johnson, M. Martin, M. Matulik and M. Utes

Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois 60510

November 1991

* Presented at the *IEEE Nuclear Science Symposium*, Santa Fe, New Mexico, November 2-9, 1991.

Operated by Universities Research Association Inc. under Contract No. DE-AC02-76CHO3000 with the United States Department of Energy

The Design of a Hardware Testing System for the D Zero Detector

R. Angstadt, M.E. Johnson, M. I. Martin, M. S. Matulik, M. Utes
Fermilab[1] P.O. Box 500 Batavia, IL 60510

Abstract

Testing a system as large as the D Zero data acquisition system is difficult. This paper describes the use of IBM compatible personal computers in a hardware test system that can run on any size system from an engineer's test bench to the entire subsystem in the D Zero Detector. The test system uses a PC to VME bus interface for the local testing and the Token Ring network for more global testing. This system has been implemented for several different hardware systems in D Zero.

I. INTRODUCTION

The D0 Detector (D0) has several computer controlled electronics systems. These include an 8000 channel Flash Analog to Digital Converter (FADC) system for the central tracking system, a sophisticated experimental timing system (CLOCK) and a 1000 channel precision High Voltage system. All of these require software for engineering development, board testing and repair, and system debugging. The engineering and test requirements can be satisfied with a single crate but system testing requires multi-crate access. In addition, the engineering tools need to be flexible and easy to modify so that prototypes can quickly be converted to finished designs.

Each of the above needs could be solved by a separate program and computer. However, this would involve a considerable investment in manpower for both software development and maintenance. Instead, D0 chose to build one program for each system which meets all these needs. By carefully choosing the hardware and software tools, we were able to use the same program on single test crates or on a large networked system of crates and computers. The amount of additional work to achieve this is relatively small.

II. OVERVIEW

Most of the digital electronics in D0 reside in VME standard crates so we were able to use one hardware standard for all test systems. We chose IBM compatible personal computers because of their low cost and the large amount of third party hardware and software that is available for them.

The most direct connection between the PC and the VME crates is a commercial interface board set manufactured by the Bit3 Computer Corporation. This device can control only one VME crate at a time and requires that the PC be close to the crate. These limitations can be overcome by using the Token Ring LAN[2] which is the control path for the D0 control system[1]. This allows any PC with a Token Ring Card to access any of the electronics in the Detector. Some of our programs were debugged by software engineers located more than a mile away from the Detector.

We use both of these interfaces in D0. The Bit3 device is used for engineering development and board test stands while the Token Ring system is used for all system diagnostics. The Token Ring system provides an independent connection into the data acquisition crates that remains available even when the physics data path is not.

Borland's Turbo Pascal was chosen because it includes many important language extensions that make possible a relatively simple, fast, and economical development environment. Wide spread use of this product provides valuable support in the form of commercial libraries as well as published articles. It has an excellent overlay manager for large programs and facilitates modular design with it's support of "units". Another important feature is that it supports compile time definitions and switches that make possible the different versions (Flavors) for the Bit3 and the Token Ring. Turbo Pascal also provides excellent graphic support through Borland's Graphics Interface (BGI) which is "device independent". The penalty for all this ease of use is lack of direct source code portability across platforms.

When Turbo Pascal is used with DOS[3] and the Bit3, it has important language extensions that facilitate direct memory mapping of Pascal Data Structures into PC absolute address space. Fig. 1 shows how PC memory may be directly overlapped into VME address space via the Bit3. This coding technique greatly simplifies and largely eliminates the need for an external data base of addresses when the VME electronics have a repetitive address pattern as is the case with the FADC and High Voltage Electronics.

Where the electronics addresses are jumper selectable and non-repetitive as in the clock hardware and on the system level for different crate base addresses, mnemonics are used that correspond to DOS file names which contain the necessary addressing information. These are ASCII files so that they may be read and edited by the program users.

[1] Work supported by the U.S. Department of Energy under contract No. DE-AC02-76CHO3000.

[2] Local Area Network.

[3] Disk Operating System.

Thus DOS files are used as a database of crate names, Token Ring Nodes and crate base addresses.

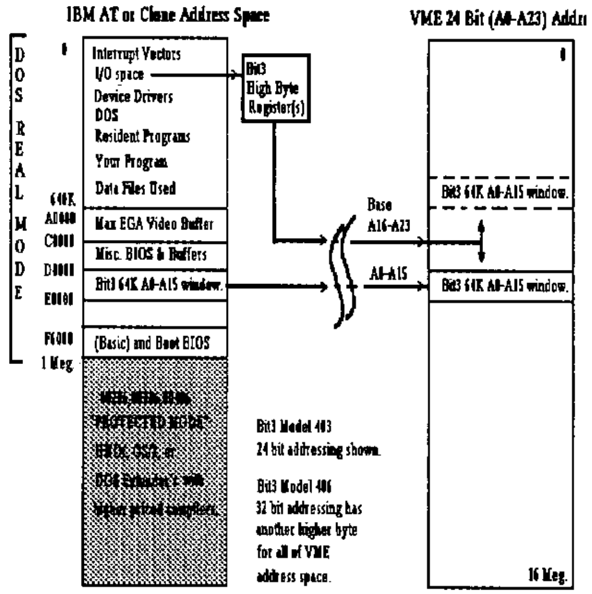

Fig. 1 Bit3 Addressing.

III. Program Details

We will describe one of the programs in more detail to further illustrate the above points. The program for the FADC system is called PCPLOT[2]. The main features of this >26K line overlaid Turbo Pascal program allows the user to:

1. Plot from 1 to 17 channels in the time domain from one FADC crate (256 channels of 100 MHZ wave form digitizers).

2. Plot 1 channel in the frequency domain with the capability to add together several waveform samples so that random noise is averaged out. This has been invaluable in debugging chamber oscillation problems and the investigation of sources of noise.

3. Change control parameters such as the length of data gathering cycle, pedestal value and test pulse amplitude.

4. Save and restore control parameters to and from disk files. (When run over the network it can do all 36 FADC crates.)

5. Save waveform data for up to one crate to disk, allowing the PC to become a small stand alone Data Acquisition System (DAQ).

6. Operate plotting and DAQ functions in a stand alone mode or parasitic to the normal D0 DAQ. That is, the same event that is read over the normal Data Acquisition System can be read on the PC.

7. Control the Shaper and Preamplifier Pulser[3] Diagnostic system over the Token Ring LAN.

8. Construct files that read and write to VME memory for custom testing or exercising the electronics without generating new code via a simple list processor. This is especially useful for development when a custom program is not yet finished. Many new boards were first exercised and tested in this manner.

9. Minimize amount of typing by using hot[4] function keys and sub-menu picks. Macro support is available for automating repetitive tasks. The program state may be customized by the user and interactively saved and restored. The program is engineered so that a menu key or button does one thing or a common sequence. The buttons may be quickly mixed and matched to fit the situation similar to Borland's IDE but without overlapping pull down menus and without re-assigning the function keys. Program variables that the user is allowed to change are displayed on the same line as the menu pick so the user can immediately see if they need or want to change it.

Single key commands can initialize a crate completely or check if a crate is responding without writing over previously downloaded values. As mentioned above a user can customize their program with initialization files and macros.

This style of program requires the user to be slightly more aware of some details of hardware and program operation, but it gives much more back in terms of its versatility, usefulness, lifespan, and utilization of both the users' and programers' time. The same program that is used to develop and test the hardware on the Engineer's bench can be used to test the hardware in-situ.

Using Borland's Turbo Pascal stand alone compiler and a Gateway 2000 80486 running at 33Mhz PCPLOT's >26K line source is ready to run in 20 to 25 seconds. Using the Integrated Development Environment (IDE) instead takes about 36 seconds. Typically one is working on a few program modules of perhaps two to four thousand lines which only take 6 to 10 seconds before one can quickly test the new code.

This rapid response makes a very effective development environment. As an example the program that was originally used to develop and control the D0 CLOCK[4] hardware originally worked with a Bit3 to one crate only. Without modification there would be some CLOCK VME crates that could not be easily tested in-situ. Also it was limited to only 25 Selector Fanout[5] modules by internal arrays and program design. Since the CLOCK hardware fills several VME crates located on different floors it was necessary to make this program work over the Token Ring to provide in-situ testing.

In less than one month it was modified so that there were both Bit3 and Token Ring Flavors similar to PCPLOT. Fig. 2 shows the different Flavors and some of their main distinctions. At the end of four months the 25 Selector Fanout limitation was designed out; the ability to plot the CLOCK timing (Timelines) was added, and loading the Timelines was made symmetrical so that the PC could be

[4] The program takes relatively immediate action when a key is pressed. There is no need to press the "Enter" key.

[5] This is one of the types of modules or cards that make up the D0 CLOCK. It is responsible for selecting and shipping out an individual Timeline to other Detector electronics.

used as a backup to the Central Computer. Originally this program was ~7K lines. By adapting and re-using a lot of code from PCPLOT the program grew to its present size of ~18K lines. This program was up and running before the VAX[6] software and was used in checking the VAX software operation.

Interface Flavor	Use
Uses Bit3 program must poll for VME DTACK	Can use with many slave crate(s) via Vertical Inter.
Uses Bit3 VME DTACK must return in 15 us.	Timeout restricts Bit3 to be in same crate as electronics.
Uses Token Ring	Anywhere LAN goes and Vertical Interconnect exists.

Fig. 2 Program Flavors.

Similarly, the engineering development program for the precision High Voltage Supplies, HVD[5], has been converted to this interface independent style in a similar amount of time. Another program, HV[6], with the cabability to support arbitrary groups of supplies across multiple slave crates is yet to be converted. These programs are discussed further in another paper presented at this conference.[7]

IV. HARDWARE CONFIGURATION

Fig. 3 shows the minimum hardware configuration necessary to run any of the programs. It consists of a PC with a Bit3 board in it connected to a VME crate which contains the other half of the Bit3 board set.

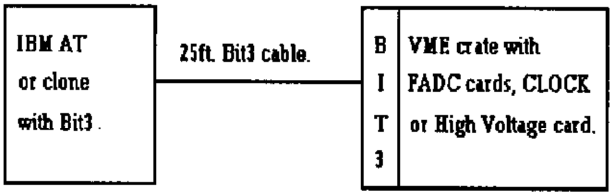

Fig. 3 Minimum Hardware Required.

It is worth mentioning that it is possible to work on and run software before the hardware is ready by running to any commercial memory board in the VME crate instead of the real hardware. This adds somewhat to the coding difficulty but it speeds up software development. A large part of the CLOCK software was converted this way and then verified on the real hardware later. Final testing and some re-working will inevitably need to be done when the real hardware becomes available.

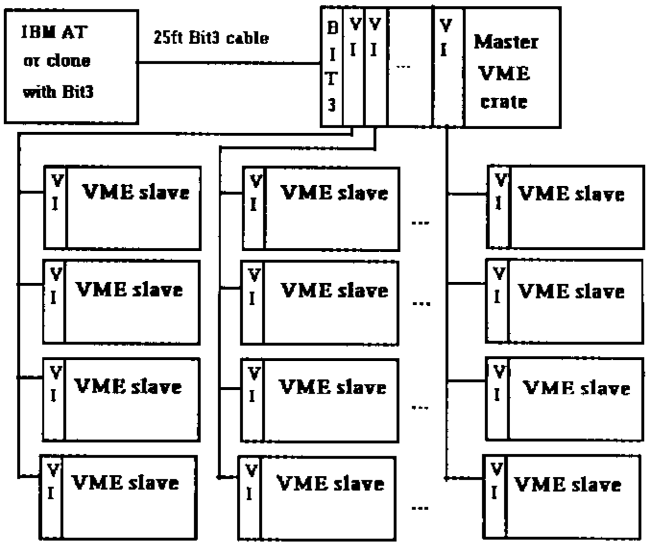

Fig. 4 Vertical Interconnects connecting many physical VME crates together into one logical addressable crate.

Fig. 4 shows how multiple physical VME crates are electronically connected through a Fermilab designed Vertical Interconnect (VI). Each slave crate has 16 Megabytes of address space. With a Bit3 in the Master VME crate the program can communicate to any one of the slave crates by changing the program Base address. This can be done interactively from a menu pick or via a mnemonic that is really a filename the user types in that follows a standard naming convention. The file contains the addressing information the program needs to communicate with that slave crate.

The Bit3 card is not restricted to the top Master crate. It may also be inserted in a slave crate to isolate problems. The maximum distance of the VI cable is 65 feet. In practice, a maximum of seven VI cards are placed in the Master VME crate although there is room for fifteen.

One of the most complex configurations is the FADC system. Fig. 5 shows a typical D0 Control Node with two of the many FADC crates connected to it via the VI (shown by the heavier black line). The heaviest line represents the signal paths from the Shaper Rack to the input of the FADC cards. The dashed line shows different possible Bit3 insertion points although normally one is left in the Master or Control crate. One Flavor of the program is used with the Bit3 and another Flavor is used with the Token Ring. One PC may have both a Bit3 and a Token Ring Card in it allowing the user to run any Flavor.

The Rack Monitor provides Digital control and readback of the Shaper hardware and Pulser Diagnostics for the FADC crates. It is only controlled through the Token Ring. Shaper control is an optional part of PCPLOT that may be enabled at run time by the user if desired.

Not shown in Fig. 5 for reasons of clarity is the high speed DAQ path from the VME Buffer Driver (VBD) back to the VAXs. There may be up to 16 FADC cards of 16

[6] VAX is manufactured by Digital Equipment Corporation.

channels in an FADC crate. "Control" is short for FADC (Trigger) Controller Card.

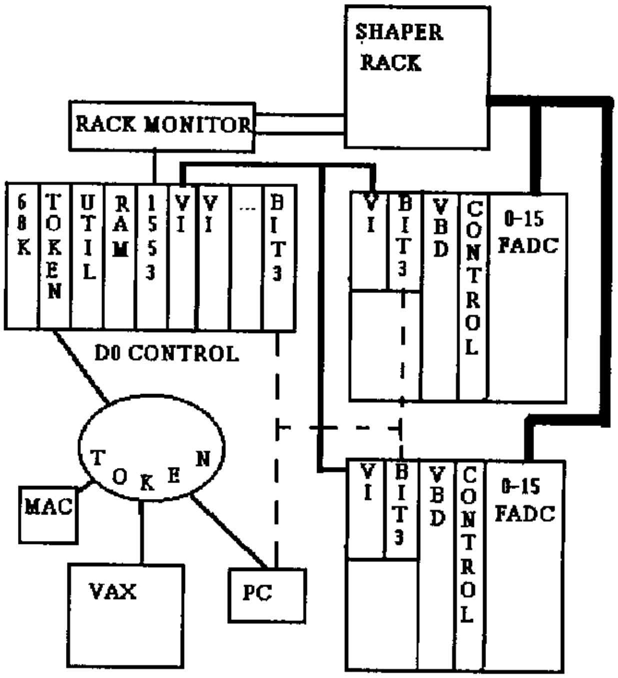

Fig. 5 PCPLOT Interface Paths .

V. Conclusion

With careful program design and a fast development cycle a small number of programers can produce large, integrated programs that satisfy both the stand-alone test station and also work over a LAN to test out the hardware in-situ. One program does it all. The users are encouraged to set it up the way they intend to use it and save the program state in a file of their own or in the program initialization file. Thus the program can be quickly customized by the user. This saves maintaining endless small changes for different customers and updating different versions when changes are made. All Flavors are current when a new feature or change is made because all Flavors come from the same body of source code.

VI. References

[1] R.W. Goodwin, R.J. Florian, M.E. Johnson, A.A. Jones and M.F. Shea, Int. Conf. on Accelerator and Large Experimental Physics Control Systems, Vancouver, BC, Canada, 1989 Nucl. Instr. and Meth. A293 (1990) 125.

[2] R. Angstadt, PCPLOT, A PC-Based FADC Control Program, D0 Note 814 (unpublished).

[3] R. Angstadt, B. Chase, B. Fellenz, M. Johnson, M. Martin, M. Matulik, G. Saewert, M. Utes, D. DeGroot, D. Voy, A Working VME based 106Mhz FADC Data Acquisition System for the Tracking Detectors at D0, these Proceedings, (IEEE Nulear Science Symposium,1991).

[4] R. Angstadt, D0 CLOCK Version 3.02, D0 Note 1259 (unpublished).

[5] R. Angstadt, G. Dougherty, HVD (High Voltage Display) Program for the PC Version 2.28. D0 Note 1257 (unpublished).

[6] M.-J. Yang, D0HV User's Manuel, D0 Note 1175 (unpublished).

[7] S.-C. Ahn, R.D. Angstadt, T.F. Droege, M.E. Johnson, B.A. MacKinnon, S.E. McNulty, M.F. Shea, R.N. Thompson, M.M. Watson, M.-J. Yang, A New VME Based High Voltage Power Supply for Large Experiments. these Proceedings, (IEEE Nulear Science Symposium,1991).