FERMILAB-Conf-92/300

Integrated FASTBUS, VME and CAMAC Diagnostic Software at Fermilab

J. Anderson, R. Forster, J. Franzen and N. Wilcer

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

October 1992

Presented at the *IEEE Conference on Nuclear Science*,
Orlando, Florida, October 27-30, 1992

Integrated FASTBUS, VME and CAMAC Diagnostic Software at Fermilab

J. Anderson, R. Forster, J. Franzen, N. Wilcer

Data Acquisition Hardware Group
Online Support Department
Computing Division
Fermi National Accelerator Laboratory
P. O. Box 500, M/S 234
Batavia, IL 60510 USA

Abstract

A fully integrated system for the diagnosis and repair of data acquisition hardware in FASTBUS[1], VME[2] and CAMAC[3] is described. A short cost/benefit analysis of using a distributed network of personal computers for diagnosis is presented. The SPUDS (Single Platform Uniting Diagnostic Software) software package developed at Fermilab by the authors is introduced. Examples of how SPUDS is currently used in the Fermilab equipment repair facility, as an evaluation tool and for field diagnostics are given.

1. Introduction

The ever increasing complexity of high energy physics data acquisition systems requires that the tools used to build these systems become more powerful. This axiom, combined with the ever-increasing cost of labor, requires that the tools become not only more powerful, but simpler to use. Physicists, engineers and technicians can ill afford long learning curves before new modular electronics become useful portions of data acquisition systems. Modern data acquisition systems use a wide variety of standards: FASTBUS, VME, CAMAC and NIM[4], to name but a few.

Methods of integrating modular electronic devices in to a working system typically require that three similar but distinct pieces of software be developed for each module:

1) Evaluation, to see if the module meets it's specifications and also to check that those specifications meet the needs of the current experiments;

2 Diagnostic, to check that all functions of the module are performing according to specification after some period of use; and

3) Repair, to provide the technician with the tools to isolate faults in a minimum of time.

Generally, evaluation software gives rise to pass/fail diagnostic software, which evolves towards a repair suite. The ideal solution to control the evolution of these various programs would be a consistent interface, library and tool set that is sufficient to meet the different criteria imposed by all three environments. In addition, this generic solution must be sufficiently affordable to be within the reach of individual university labs.

Recent collaborations have spawned a number of small stations at universities, whose budgets cannot afford expensive platforms. Recent advances in personal computers have made these machines sufficiently powerful to use them as the basis of diagnostic or evaluation systems. The software tools on personal computers are more than sufficient to design high quality user interfaces, leading to a solution based upon PC's[5] rather than traditional minicomputers.

Although these three needs have existed for many years, the historical solution has been to separate them. We have defined and implemented a system, based on personal computers, that ties these three tasks together to achieve increased productivity in all of them. Our solution is a software system named SPUDS (Single Platform Uniting Diagnostic Software).

2. A cost/benefit analysis of the use of PC's

In the repair and evaluation environments, a PC is significantly more economical than previous solutions. PC's now boast processor power sufficient to perform any reasonable diagnostic, and in fact PC's can even run small (one-rack) data acquisition systems. PC's are found at most labs and universities already. With sufficient power, lower cost and easy upgrades, PC's are a logical choice for test stands.

PC's are also easily integrated into networks. When combined with file servers PC's form a formidable distributed computing system. The vast amount of low- or no-cost software from the shareware market further increases their value. Typical PC system costs are shown in Figure 1. Cost comparisons to alternate test rack configurations are left to the reader.

Figure 1: PC with SPUDS

Item	Cost
PC-486 system with keyboard, CPU, memory, and color monitor	$3000
DOS	$30
LeCroy 1691A PC-1821 interface	$650
LeCroy 1691A PC-1821 cable	$20
LeCroy 1821 Segment Manager/Interface	$9,000
DSP6002/6004 Crate Controller/Interface set	$3000
Interface cable	$30
MicroSoft Quick C[1]	$125
SPUDS	$0
Interface subtotal:	$12,700
Total:	**$15,855**

[1] Code development software with built-in editor and source-level debugger

3. An Introduction to SPUDS

SPUDS is a graphical user interface combined with libraries and utilities to allow the programmer or technician to easily develop programs for CAMAC, FASTBUS or VME use. SPUDS integrates a run-time menu manager, subroutine library and utilities package with on-line help and development assistance. Modular design allows the upgrade or enhancement of any portion of the system without adverse effect upon previously designed code. Support for multiple development languages is designed in from the start, with the current version capable of supporting both C and FORTRAN development. A special FASTBUS assembly language named SONIC[6], endemic to the chosen FASTBUS interface, and a BASIC-like CAMAC language, PCCDL, are also provided. Other languages are expected to be added as needed; the system has slots provided for BASIC and PASCAL, plus one "Custom" slot for whatever the user desires.

SPUDS is designed to be used on any member of the PC class of computers. The suggested minimum system is:

- An 80286-based machine with 640K of memory;
- A 640 X 350 EGA graphics display;
- sufficient expansion slots to hold the external bus interfaces.

SPUDS is both a user interface and a development environment, with customizations applied by the user. This customization of the product is handled through ASCII control files, allowing a technician to graduate to code development at any time. Further, the development tools needed to create software under the SPUDS umbrella are also not fixed; new compilers may be installed either in addition to or in replacement of existent languages. The user may integrate in whatever text editor is desired, and this editor may be changed at any time.

As distributed, SPUDS comes complete with user interface, subroutine library, on-line help, include files, and utility programs for FASTBUS, CAMAC and VME. The user is responsible for providing any compilers or text editors, which are only required for code development. A number of test packages for commonly used FASTBUS and CAMAC modules have already been written for use under SPUDS, and these will run under the program as distributed without modification.

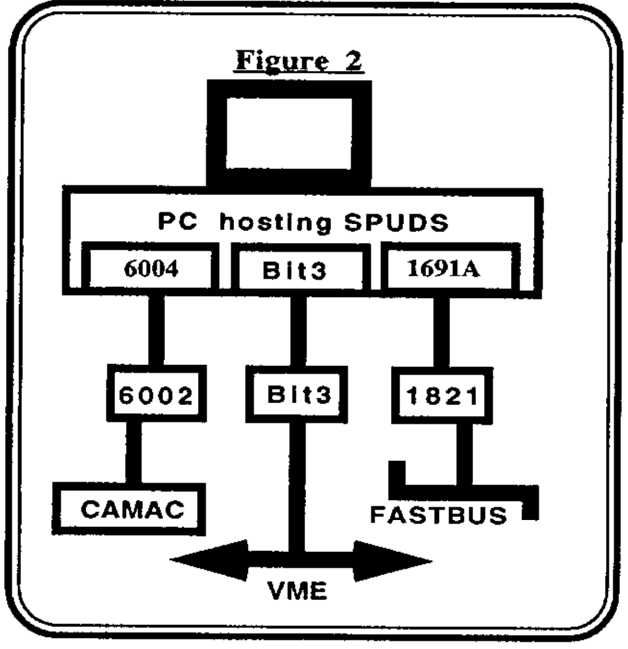
Simple menu structures plus judicious use of the PC's function keys make SPUDS easy to learn. A set of function keys is reserved for user-defined activities. A fixed directory structure allows simple control of backup and distribution. The SPUDS system has been tested in a network environment at Fermilab, where the Data Acquisition Hardware Group uses Ethernet (DECnet) to distribute SPUDS programs and software updates to the Equipment Support Department regularly.

SPUDS connects to the external busses via a set of commercially available interfaces. The interfaces chosen are based upon simplistic criteria of general availability, sufficient operational speed and intercompatibilty. Other interfaces could have been used; the selections made do not constitute an endorsement of these particular units by Fermilab, URA[7] or the United States Government. The interfaces used are:

FASTBUS:	The LeCroy 1821 and it's 1691A interface.
CAMAC:	The DSP[8] 6002 and it's 6004 interface.
VME:	The Bit3[9] pair of interface cards.

A schematic representation of the PC connections to each of the busses is shown in Figure 2.

Figure 2



The SPUDS software is not dependent on having all interfaces present. Only those which are needed for the task at hand must be installed. Procurement and installation of these interfaces is the responsibility of the user; we only supply the software. A Setup function with SPUDS allows the user to customize the software to match the hardware installation.

SPUDS offers a superior method for developing diagnostic, repair and evaluation software in a research environment. The SPUDS system is designed to provide a highly integrated set of functions in the most commonly used bus systems at a cost much less than that of previous solutions. By tying a number of external busses to a single environment at low cost, SPUDS allows the use of commercial modules as test units and makes each repair station capable of servicing a wider range of devices.

Those readers familiar with LeCroy's LIFT user interface will recognize the similarities between LIFT and SPUDS. SPUDS was in many ways inspired by LIFT, but is an independent creation. Fermilab has purchased a copy of the LIFT library source, which was used as a functional model for the SPUDS libraries; however, although the names of many functions are the same to ease conversion, the code within the library is quite different. Conversion from LIFT to SPUDS is eased by a utility provided in the SPUDS distribution. Questions about the SPUDS software should be forwarded to the authors of this document either by surface or electronic mail[10].

4. Experience using SPUDS in Equipment Support

Integrating the most commonly used bus systems in to a single platform provides a number of benefits to the repair facility. A common user interface allows the new technician to be trained more quickly - instead of having to learn hardware and software for every new procedure, the software is learned once for all. In addition, the reduced cost of the SPUDS system allows each test setup to become more generalized, providing the repair manager with enhanced flexibility. Lastly, a distributed system of test computers has the advantage that a computer failure disrupts work only in one place, and not throughout the department.

At Fermilab, SPUDS and other PC-based diagnostics have been installed as part of a program to phase out dependence upon our obsolete DEC PDP-11 computers for repair. The transition has been smooth, with the technicians involved with the Data Acquisition Hardware group during the entire process. Larger, more complex diagnostic and repair suites were recoded by the DAH group, but a much greater number of smaller programs have been converted by the technicians themselves.

In addition to SPUDS, we have also installed network services that allow the PC's in the Equipment Support Department to connect directly to a file server which contains all the master copies of the software packages. Technicians can with one or two DOS commands download entire test suites, making software updates painless. The network also provides each technician access to every test program set, increasing flexibility.

SPUDS interfaces the common modular busses into a environment providing a standardized "look and feel" to all levels of instrument repair tasks. Test stand hardware costs are minimized since modules from any or all of the supported busses can be involved in any specific test. Multiple versions of most custom or specialized test and calibration modules are no longer necessary since any test can access any required and supported bus, further reducing the test equipment investment.

As SPUDS has developed, a number of benefits from the new system have become apparent:

1) The technicians are now becoming able to design their own special loops and analyzer setups using the utilities provided - they are no longer constrained by the limits of a prepackaged diagnostic.

2) The development time for new versions of software has been cut dramatically due to the modular nature of

SPUDS diagnostics. Rather than rebuilding a large, monolithic program, new features can often be added in hours by just adding one more small program to the existent set.

3) The transition time from evaluation software to repair software is greatly decreased. Since the same platform is used for everything from evaluation to repair, the required facilities to perform repair can be built in to the software from the start.

4) The network provides the required backup so that the technicians may freely be given the source code for everything. This, plus the on-line help system, allows them a much deeper level of understanding of what each piece of code is doing.

5) SPUDS provides an easy means to certify the performance of a target module using a user-defined sequence, or gauntlet, of Go/NoGo (Pass/Fail) tests. Execution of such a sequence terminates when an error is detected, or when the sequence is completed, or even when the entire "gauntlet" of tests has been repeated successfully a predetermined number of times.

The consistent interface allows different levels of technician to use the same software in different ways. Gauntlet mode testing is appropriate for use by computer science interns, students, or even our "Least Technical" technician. Such testing divides incoming modules into "Good" modules suitable for reissue, and "Bad" modules needing the ministrations of a "Technical" technician.

"Technical" technicians appreciate the detailed library error messages available to properly written SPUDS diagnostic test code. Additionally, SPUDS test developers are encouraged to add as many detailed error messages as desired; multiple functions are provided within the library to display said messages in a consistent manner. The on-line help system integrated in to the package allows the software developer to provide menu-driven help for each facet of a software package; with use of widely available graphics tools, pictorial help may also be built in.

5. Experience using SPUDS in Evaluation

The evaluation process differs from the repair process in both degree and mandate, emphasizing completeness and accuracy over execution time. For example, Differential Non-Linearity tests must be performed across the entire input range during evaluation testing, searching for evidence of inflection points. In the absence of unusual results, it's then reasonable to select several test point values based on those results for the quick repair test. The SPUDS environment makes it easy to modify such typical evaluation tests into the desired repair test. Note that the converse is just as true, should reevaluation or special case testing ever become necessary.

Frequently, an evaluation test is merely a repair test executed using extended parameters. For instance, a full-scale ADC or TDC range might be divided into more smaller steps. Once a characteristic performance curve is determined, repair tests can use fewer larger steps to complete the test faster. The systematic study of certain performance characteristics is usually reducible to testing a few key test point values.

Other evaluation tests may simply not be necessary for repair purposes. An evaluation test typically will ask the question "does the unit do everything it's supposed to, and does it also not do anything it should not do?". The repair software will often cease after the first half of the question. Evaluation software usually performs a greater number of repetitions to give more accurate statistics, which then are used to determine boundary conditions which are tested by the diagnostic or repair software.

The inclusion of a series of utilities to perform protocol-independent primitive functions (BUSMON, PCCDL, SLAVE, EVX, BIT3BUG) frees the software developer to concentrate solely on the unique functions of the device under test. This being the case, repair software does often reduce to a subset of evaluation software; one need not add special loops or setups to assist the repair technician in problem isolation because those functions are already present in the utilities.

Complex code need not be developed to evaluate a piece of hardware. The SLAVE program supports a very simple meta-language that allows FASTBUS sequences to be defined in as little as one line of code. PCCDL and BIT3BUG are no more complex than BASIC. BUSMON requires no programming ability at all; it is a "point and shoot" FASTBUS control program. Once the base algorithms are tested in these utilities conversion to a program is performed by looking up the equivalent subroutine calls in the SPUDS library to the utility functions. A nearly one-to-one correspondence exists from utility functions to subroutine calls.

Since modern data acquisition equipment is designed for maximal functional speed obtaining low-level diagnostic information requires utility software capable of performing the protocol primitives of the respective busses. The very first step in evaluation or repair is to determine that the device in question can perform the most rudimentary functions. Low level tools typified by the BUSMON Fastbus utility allow users to drive individual bus lines, without imposing any bus protocol requirements; this methodology allows the evaluation of the response of the module to both legal and illegal protocol.

6. Experience using SPUDS in field diagnostics

The introduction, subsequent downsizing and weight reduction of portable PCs, coupled with the development of the SPUDS environment, has provided the ability to outfit a technician with the tools required to test a Fastbus/Camac/VME system in the field. Consider the case of Fastbus trouble in the field. Previous Fastbus troubleshooting methods were typically limited to module swapping. The swap method was hindered if spare modules were not immediately available in the field.

Pinpointing trouble has usually been limited to deductive reasoning. Although it is still the first measure of problem analysis, deductive reasoning is often not enough to definitively isolate a problem. Diagnostic software can often isolate or verify a fault. A technician can now visit the field carrying all the tools required to diagnose faults down to a replaceable module, and occasionally, to a replaceable component. The field technician will likely exercise the same diagnostic tests which are used by the in-house repair facility.

The SPUDS SETUP menu facilitates real-time hardware configuration changes. Well-coded SPUDS tests don't require recompilation to migrate to different test stand configurations.

This increased functionality in the field has allowed Fermilab experiments to reduce the number of "false failures" - that is, the number of correctly functioning modules which are returned to the repair center as suspected failures. Elimination of unneeded repair cycles can both increase the efficiency of the repair department and decrease experiment downtime. A reasonable set of PC-based diagnostics can easily identify the culprit in the wide majority of digital problems, and can often provide "confidence tests" of analog data. Of course, field diagnostics have their limit - it is usually impractical to test differential non-linearity of an ADC without significant external equipment, which is only available at the test bench.

The greatest benefits occur, of course, when the readout controller used by the experiment is the same as that used by the diagnostic system. The next best case occurs when slots are reserved within the data acquisition crates for diagnostic modules, and the bus system provides for multi-Master arbitration. In either case, inserting the diagnostic system in to the data acquisition becomes easy, and the biggest problem is then the unfamiliarity of the computer system used by the experiment.

The minimal cost of PC's conquers the learning curve by either allowing the technician to bring the computer along (via a portable) or by designing in a PC as part of the rack structure. With use of commercially available networking or remote login software, the PC can be integrated into the front end at the beginning and diagnostics run from the relative comfort of the counting room. The technician is then assured of a known software environment independent of the structure of the experimental computer system, greatly accelerating the diagnosis process. Further, with known tools, setting up scope loops or logic analyzer triggers is greatly facilitated.

Summary

Low cost, highly reliable systems for diagnosis and repair are easily achievable using a distributed network of personal computers. With the advent of recent 80486 technology, a low-cost personal computer can provide at least equivalent, potentially superior repair turnaround time as compared to shared use of more costly minicomputers. The introduction of the SPUDS software shell provides a consistent framework for the development, maintenance and updating of diagnostics that is usable by collaborations, individual universities and vendors. SPUDS has become the platform of choice for diagnostics and evaluation at Fermilab.

As distributed, SPUDS provides a library containing a wide array of functions to allow user programs to exercise modules, plus a series of user-friendly utilities for exercising FASTBUS, CAMAC and VME. Data from all external busses is easily ported to analysis packages. The SPUDS library uses a PASCAL calling sequence to ease the use of multiple languages, including PASCAL, FORTRAN and C. An alternative form of the library with a C calling convention is also provided.

The consistency of the SPUDS system across the spectrum of diagnostic work, from evaluation of new products to diagnosis of problems in the field, increases the base level of understanding of data acquisition systems while shortening the learning curve. SPUDS provides the same functionality achievable with the combination of the LIFT diagnostic package (copyright LeCroy Corporation), the FASTBUS standard routines and the well-known CAMAC and FASTBUS Diagnostic Languages (CDL and DDL), but with compete integration. SPUDS yields higher productivity than the set of the individual tools, proving again that "The Whole Is Greater Than The Sum Of Its Parts".

Notes

1. FASTBUS is ANSI/IEEE STD 960-1986.
2. VME, for Versa Module Europa, is IEEE 1014-1987.
3. CAMAC, for Computer Automated Measurement And Control, is described electrically by IEEE STD 583 and mechanically by IEEE STD 596.
4. NIM, for Nuclear Instrumentation Module, is described in AEC Report No. TID-20893.
5. The generic term "PC" refers herein to any "Personal Computer" of the style pioneered by International Business Machines, IBM.
6. SONIC, for Sequencer ON-line Interactive Code, originally from LeCroy, 700 Chestnut Ridge Road, Chestnut Ridge, New York 10977-6499.
7. URA, for Universities Research Association, is the sole funding agency for Fermilab.
8. DSP, for DSP Technology 48500 Kato Road, Fremont, CA 94538-7338.
9. Bit3 Computer Corporation, 8120 Penn Avenue South, Minneapolis, MN 55431-1393.
10. E-Mail addresses for the authors are JANDERSON (or FORSTER or FRANZEN or WILCER) @FNAL.FNAL.GOV .