"FASTBUS" - A DESCRIPTION, A STATUS REPORT, AND

A SUMMARY OF ONGOING PROJECTS

Edward J. Barsotti

Fermi National Accelerator Laboratory Batavia, Illinois U.S.A.

ABSTRACT

FASTBUS is a modular data and control bus and mechanical packaging standard currently under development. It is being funded by the United States Department of Energy and fully supported by the European ESONE Committee. FASTBUS is being designed to meet the high-speed data acquisition and parallel and distributed processing requirements of the next generation of large-scale physics experiments. It is a multiprocessor system with multiple bus segments which operate independently but link together for passing data. It operates asynchronously to accommodate very high and very low speed devices over long and short paths, using handshake protocols for reliability. It can also operate synchronously without handshakes for transfer of data blocks at maximum speed. The motivation for developing FASTBUS, its goals, history and organizational structure are summarized. Its features such as bus protocols, sparse data scan capabilities, power supplies and mechanics, bus segmentation and parallel processing structures, etc. are described. European and United States systems being proposed or built using FASTBUS along with hardware and sotware projects to aid in the standard or system development are discussed. The paper concludes by reviewing the present state of and development schedules for the FASTBUS standard.

INTRODUCTION

In the mid-1970's both the NIM/CAMAC Committee and the High Energy Physics community were simultaneously becoming concerned that future data-rate experiments would be hampered by the present CAMAC data bus speed of 1 MHz maximum and its unfavorable architecture for supporting parallel and distributed processing. A request made by members of the physics community in 1975 to the NIM/CAMAC Committee to study the above problem led to the formation of the NIM Advanced Systems Study Group. This study group's report led to the formation of the present FASTBUS design group, the NIM Fast System Design Group (FSDG). Preliminary draft documents of the FASTBUS, specification have been distributed by this group at regular intervals. 2,3,4,5,6,7,8,9 From the beginning European ESONE members particularly those from CERN have attended FSDG meetings and contributed very substantially to the development of the FASTBUS standard. Presently several ESONE member institutions are writing papers, performing tests, and thoroughly reviewing FSDG documents at ESONE Advanced Systems Design Group meetings.

As of this writing, a subgroup of the FSDG is preparing an update to the latest draft specification which will add substantial technical clarification to the previous document and try to make the new document technically accurate and complete. Additionally, an enormous amount of prototyping work is currently in progress. SLAC and the University of Illinois are the primary institutions doing the massive amount of work necessary to test the integrity of the system as well as provide potential users with support hardware and software (e.g., crates, kluge cards, ancillary logic modules, system initialization software, Segment Interconnects, etc.). Other institutions are planning or heavily working on FASTBUS data acquisition and processing systems. Both the support work and the system development work will be reported in this paper.

Several prior papers have described the details of the FASTBUS draft

specification. Two of the most recent papers ^{10,11} summarize hardware prototyping efforts and software status as of November 19. This paper will summarize and update the description of FASTBUS and then concentrate on the details of the FASTBUS specification support hardware and software efforts and development efforts and schedules of FASTBUS systems being worked on in Europe, Canada and the United States. In conclusion, the paper reviews the development schedules of the FASTBUS standard and lists the NIM/FASTBUS organizational structure and European, Canadian, and United States participating institutions.

The FASTBUS systems design goals as stated in an excellent FASTBUS overview paper 12 by B. Wadsworth are:

Highest possible speed - at least an order of magnitude bandwidth improvement over CAMAC - to cope with the high data rates from large complex particle detectors and to allow the highest possible trigger rate.

Variable speed, allowing both asynchronous handshaked transfers between different speed devices and synchronous non-handshaked transfers for maximum block transfer speed.

Structured to accommodate simple, fast scanning of sparse data from large detector arrays.

Accommodation of fast modular pre-processors at the subsystem level - to reduce the quantity of data which must be transmitted through the system and recorded in high data rate experiments.

Bus segmentation - to facilitate parallel processing at the subsystem level.

Protocol - uniform system wide and not based on a specific circuit technology. One which includes the ability to perform macro and broadcast commands.

Large address space - to accommodate very large semiconductor memories which offer the possibility of high speed computation by means of look-up tables.

Logical addressing - a module's address should be independent of its physical location on a segment.

Modular construction — to allow for the development of standard function modules which can be re-used in a variety of configurations; to encourage outside manufacturing support; to permit custom designed equipment to be included in a system in a cost-effective manner.

The following summary of the FASTBUS specification will show that these goals are being met. Technology independence has additionally been achieved. (i.e., current technology dictated ECL logic be used on the bus to meet the speed requirements but future technologies can easily be incorporated into the FASTBUS architecture).

Since much of the following tutorial has been well stated in previous papers, much of the material has been extracted from these earlier papers.

TOPOLOGY

The analog to the CAMAC Dataway and the Crate Controller in FASTBUS is the Segment and the Segment Interconnect (SI) respectively. Figure 1 shows a FASTBUS data acquisition system. The power and thus usefulness of FASTBUS over CAMAC is in the dissimilar functioning of Segments and Segment Interconnects over their CAMAC counterparts. Arbitration logic exists on every segment such that several master devices can reside on a segment and

communicate to any other segment through SI's. Segments or clusters of segments can operate totally asynchronously and independent of simultaneous operations on other segments. 32-bit single transfers can occur in roughly 100 nanoseconds within a crate segment. Block transfers on that segment can occur at a rate of one 32-bit word every 40 nanoseconds when necessary. The cable segment obeys the same protocol as that on the crate segment and thus can itself have several master and slave devices connected to it.

PROTOCOL

Table 1 lists the signal lines required to implement the FASTBUS protocol. FASTBUS uses 32 lines in a multiplexed mode for address and data.

TABLE 1 FASTBUS SIGNALS

	r Ac	TBUS SI	GIAND		
Mnemonic	Signal Name	-	Use	No.	Comments
AS	Address Sync		T/C	1	for addressing
AK	Address Acknowledge		T/C	ī	and reporting
EG	Enable Geogrpahical		-/ C	ī	
LG.	mable deogrpanical		•	-	connection
					Connection
СВ	Control/Block		C	1	
NH	No Handshake		Ċ	1	
RD	Read		Ċ	1	
AD	Address/Data		I	32	
BK	Busy Acknowledge		I	1	
NK	Negative Acknowledge		Ī	ī	
WT	Wait		I/C	ī	
			-, •	_	
DS	Data Sync		T	1	for data and
DK	Data Acknowledge		Ť	ī	control of data
					transfers
PA	Parity		I	1	
PE	Parity Enable		I/C	1	
SR	Service Request		I	1	
RB	Reset Bus		С	1	
BH*	Bus Halted		C	1	
					•
AG	Arbitration Grant		T	1	
AL	Arbitration Vector		I	6	
AR	Arbitration Request		, C	1	for bus
AI*	Arbitration Inhibit	* *	C	1	arbitration
GK	Grant Acknowledge	۲.	T.	1	
GT III	Gardal Time Musuamit		T /O	,	
SLT	Serial Line Transmit		I/C	ļ	for Minampolisa
SLR	Serial Line Return			_1	for diagnostics
				60	
				90	
GA	Geographical Address	Ding	I	5	CRATE SEGMENT only
GA.	(Position encoded	FINS	-	,	(Simulated on CABLE
1	-not bussed)				SEGMENT)
	-not bussed)				SEGMENT)
TP	T-Pin (not bussed)			1	
DL	Daisy Chain Left			2	
DR	Daisy Chain Right			2	
R	Reserved			24	
1	Vesc I A Ca			47	
1	T=Timing				
1	C=Control				
1	I=Information				
	*Proposed Lines				
L					

All FASTBUS operations involve a master first requesting and being granted bus mastership. The Master then selects a Slave by an address cycle and follows this by any number of data transfer cycles after which the bus is released.

An address cycle is started by the Master asserting the Slave's address on the 32 Address Data (AD) lines followed by Address Sync (AS). The address word assertion sets up a path, through Segment Interconnects if necessary, between Master and Slave. When the Slave recognizes its address, it responds with the Address Acknowledge signal (AK). The protocol requires that AS and AK remain asserted until the operation is completed. The AS/AK lock serves to cause all other devices to ignore bus activity thus allowing the communicating pair to employ any protocol they both understand as long as the AS/AK lock is not broken. In order to facilitate the construction of compatible Devices, however, standard protocols for most useful operations have been specified.

On receipt of the AK response from the Slave, the Master removes the address information from the AD lines and uses these lines for data during the ensuing data transfer cycles. For a Write operation, the Master asserts data on the AD lines and follows this assertion by the Data Sync (DS). The Slave indicates its acceptance of the data by issuing a Data Acknowledge (DK). For a read operation the Master asserts the DS and the Read Data (RD) lines. The Slave responds by placing data on the AD lines and issuing DK which is used by the Master to latch the data. The operation is terminated by the Master removing all its signals including AS from the bus. The Slave, sensing the removal of AS, removes all its signals including AK.

Since Address and Data Cycles are easily identifiable, the Control/Block line (CB) and the No Handshake line (NH) can be used by the Master to modify the meaning of the address information and to specify the type of data transfer. During address time, control of data space can be specified as well as single or multiple listener mode. During data time, random data, extended address, or handshake or non-handshake block transfer can be specified. Table 2 lists these various operations.

Figures 2 through 9 are timing diagrams for various FASTBUS operations. Figure 9 shows how the WAIT (WT) signal line is used to disable a master's response timeouts when a Slave has a slow access time.

The Master (or SI) has the responsibility to insert delays to accommodate signal skew on a segment. The Slave is always guaranteed that AD and control signal line information is stable at the assertion of AS or at a DS transition. Slaves returning data and/or NK, BK response information need only assert that information in time coincidence with the assertion of AK or a DK transition. Tables 3 and 4 list Segment Interconnect or processor interface NK, BK responses at AK and Slave responses at DK time respectively.

TABLE 2
FASTBUS OPERATIONS

	Ycle	
	s Data I CB NH	Function
00	00 01 10 11	Specific Address (Data Space) Single Word, Handshake (HS) Extended Address, HS Block Transfer, HS Block Transfer, Non-HS
01	00 01 10 11	Broadcast (Data Space) Single Word, System HS Extended Address, System HS Block Transfer System HS Block Transfer, Non-HS
10	00 01 10 11	Specific Address (Control Space) Single Word, HS Extended Address, HS Block Transfer, HS Block Transfer, Non-HS
11	00 01 10 11	Broadcast (Control Space) Single Word, System HS Extended Address, System HS Block Transfer, System HS Block Transfer, Non-HS

TABLE 3
ADDRESS TIME BK-NK RESPONSE WITH THE ASSERTION OF AK

ВК	NK	(SI or Processor Interface) Device Response
0	0	Address Recognized
0	1	Network Failure
1	0	Network Busy
1	1	Network Abort

TABLE 4
BK-NK SLAVE RESPONSE TO A DK TRANSITION

BK	<u>NK</u>	Device Response
0	0	Standard response if a data transfer has occured.
0	1	Slave has detected error.
1	0	Slave is currently Busy.
1	1	RD=0;Slave cannot accept this or any more data. RD=1;Slave has no more data.

ADDRESSING

Two basic modes of addressing are implemented in the FASTBUS standard, the recommended or logical addressing mode and the geographical addressing mode. System initialization (e.g., logical address loading) must be done by geographical addressing. Logical addressing is independent of a devices' physical position in a (crate) segment. Definition of fields in a logical address is given in Figure 10. The modules' device address (DA) is comprised of the group field (GP) and the module address field (MA). The GP field is used by SI's to pass messages. Most all modules have identical GP fields on a given segment. A Slave's MA field is the primary field used to select only one device during an operation. Note that modules with unusually large address space requirements can have their MA field extend into their GP field. The IA field is used to select different parts of the same device. Thus a 16-channel scaler would require two bits in its IA field.

A device must fully decode all 32 AD bits during an address cycle as well as during an extended address cycle if implemented.

Geographical Addressing Implementation

In a crate segment, Geographical Addressing is accomplished with circuitry similar to that shown in Figure 11. Each crate segment slot has its five GA pins wired (in the backplane) to the binary value of the slot position. Slot 0 has GA (04:00) equal to 00000, slot 1 has GA (04:00) equal to 00001, etc. When a devices' GA signal lines compare with AD (04:00) and AD (07:05) are zero at the assertion of EG, the module is geographically addressed. Geographical addressing on a cable segment is only slightly different in that every device on a cable segment must additionally have five switches to be used in place of crate segment GA pins.

Broadcast Operations

In a Broadcast operation, a Master sends information to multiple Slaves. Broadcasts can be used for various purposes, such as synchronizing Devices or clearing a bank of counters. Since more than one Slave may be involved, no meaningful handshake between Slave and Master is possible. However, a System Handshake from ancillary logic on the last segment of any route informs the Master that its command has propagated to every Segment to which it was addressed.

In Segment Interconnects, the route table entries corresponding to the lowest GP field address of messages passed is used for routing Global Broadcasts. More than one Segment Interconnect may recognize and pass a Broadcast message since no individual Handshakes are returned. The pattern formed by a propagating Broadcast message must be a simple tree with no cross connections. This rule has to be enforced by the initialization program.

When the Broadcast address has propagated successfully throughout the system, the System Handshake occurs and the Master asserts the information it wishes to Broadcast. Multiple Data Cycles follow as needed, using the System Handshake. Thus, any kind of standard Write Operation may be performed simultaneously on a set of Devices by a Broadcast.

If NH=1 at the assertion of the AS signal, the AD lines are interpreted as a Broadcast Address as shown in Figure 12. If L=1 $\,$

devices respond depending upon X, T and the condition code. X=1 enables device detection of the condition code. T=1 conditionally permits devices to assert TP on the following read cycle or respond if their TP is asserted on the following write cycle. The G bit is used by SI's along with their routing information to determine whether to pass a Broadcast message. T, X information is used to determine responses as shown in Table 5.

TABLE 5
SUMMARY OF T AND X USUAGE FOR BROADCAST OPERATIONS

T	<u>x</u>	Device Response
0	0	Unconditional; all Devices respond without regard for TP or condition
0	1	Categorical; only Devices of Category N (specified during Device initialization) respond
1	0	TP; Sparse Data Scan or Pattern Select, use TP during next cycle for reporting or selection
1	1	Conditional; Devices respond depending on condition specified

Broadcast operations are very flexible. They may be sent to the local segment only, to a local segment and routed segments below the local segment, to only routed segments below the local segment, to a specific segment or to a specific segment and then to routed segments below that segment. Broadcast operations as shown in Table 5 can be conditionally directed to modules by using the TP pin, by selecting catagorical devices or by using the condition field.

Control and Status Space Addressing

The recommended method for accessing Control and Status Register (CSR) space uses two cycles of addressing. This method may be used to select either a single Device or, by Broadcasting, a group of Devices. The first cycle is an Address Cycle with CB=1 and is used simply to select the Device(s). The second cycle is an Extended Address Data Cycle in which the 32-bit Address of the register in CSR space is asserted on the AD lines by the Master along with CB=0, NH=1. This cycle may be omitted if CSR register 0 is to be accessed or if the CSR register Address can be contained in the IA Field of the Device. Once the addressing is complete, Data Cycles (single or block transfers) can follow.

So that there would be symmetry in the method for setting and clearing bits in CSR registers during both normal and Broadcast operations, the following was adopted. Each operation in a register required to be selectively set or cleared has both a "set" (enable) and a "clear" (disable) bit assigned to it and functioning as shown in Table 6. This implementaiton was necessary because the common method of using a Read-Modify-Write operation is not possible for Broadcast Operations.

TABLE 6
CSR SELECTIVE SET/CLEAR FUNCTION IMPLEMENTATION

Set-Bit	Clear-Bit	Action
0	0	No-operation
0	1	Function bit cleared to 0 ("OFF")
1	0	Function bit set to 1 ("ON")
1	1	Undefined

The implementation of Register 0 is mandatory in every device. It has very commonly used functions (e.g., enable/disable logical addressing, reset) as well as a read field containing its identification number. It was strongly felt that requiring this register would better standardize software.

SEGMENT ANCILLARY LOGIC

Shown in Figure 13 is the Ancillary Logic which is an integral part of every FASTBUS segment. This logic contains terminations, arbitration, geographic addressing and Broadcast message handshake circuitry for the segment. On a crate segment, the Ancillary Logic is on two printed circuit cards mounted on either end of the segment on the rear of a crate. On the cable segment it again is two cards (devices) connected at either end of the cable segment. The cable segment of the standard SI under development will exit the SI at the rear auxiliary connector area. Cable segment ancillary logic will most likely be attached at the rear auxiliary connector also.

It is the Ancillary Logic which normally asserts EG during a Geographical Addressing operation. This assertion occurs when 24 leading zeros are detected in the address field at the time a master asserts Address Sync.

A current proposal adds a Segment Number Register to the segments' ancillary logic. The contents of the register is compared with the upper address bits at address time. If the comparison is true and all lower order bits to and including AD(08) are zero, EG is asserted by the ancillary logic. This modification aids in software transportability between segments while not requiring upper address field comparing and zeroing logic in every master performing a Geographical Address operation to a device on its segment.

BUS ARBITRATION

Contention for bus (segment) mastership is resolved by a part of the segments Ancillary Logic called the Arbitration Timing Controller (ATC). The FASTBUS arbitration lines are listed in Table 7. Each Master is assigned a 6-bit Arbitration Level. Masters wishing to gain bus mastership assert the Arbitration Request line (AR), wait for Arbitration Grant (AG), and then place their arbitration code on the six Arbitration Level lines AL(05:00). If the Grant Acknowledge, GK, line is not asserted, the ATC starts an Arbitration Cycle by asserting the AG signal. As Arbitration Levels are placed on the AL lines by a "wire-or" function, any asserting requestor will override all non-asserting ones at each bit position. Each requestor continully compares its AL with the code on the AL lines bit-by-bit from most to least significant positions. If a requestor notes that a bit which it did not assert is being asserted, it realizes that a higher level requestor is competing and stops asserting all bits of less

significance. After a time determined by the ATC, only the highest AL remains asserted on the AL lines and each competitor knows if it has won or lost. When the ATC has determined that the bus is free (AS=AK=WT=GK=0) it stops asserting AG and the winning Master responds by asserting GK and assuming bus Mastership. Note that arbitration can be overlapped with the ending of a previous operation if the Master release GK early.

Priority for arbitration can be assigned at one of two levels. At the local level (AL05=0), 31 vectors are assigned uniquely among devices on a given segment, and have relevance only on that segment. These are called local Priority Levels. At the global level (AL05=1), 32 vectors are assigned uniquely and serve to indicate the priority of important system— wide transactions. These are called System Priority Levels. A transaction with Local level priority may be directed to destinations beyond the local segment. However, its priority varies as it moves through the system. Each intervening SI substitutes its own assigned priority as it passes the transaction to its far-side segment.

Two compatible protocols are possible with the specified arbitration scheme. In one, Arbitration Requests are made without regard to the presence of any other requests. This leads to the possibility of low priority Masters being denied bus mastership for intolerably long periods of time. In the other proposed protocol, AR's are made only if the Arbitration Inhibit line is false. This line is set at the start of an Arbitration Cycle and reset only when no AR's are outstanding. Hence, all AR's existing at a given instant of time are satisfied before any new ones can be made.

TABLE 7
FASTBUS ARBITRATION LINES

Name	Designation	<u>Use</u>
Arbitration Request	AR	Generated by a Master to request Mastership
Arbitration Inhibit	AI*	Generated by ATC to indicate presence of unsatisfied requests
Arbitration Grant	AG	Generated by the ATC to control the Arbitration cycle timing
Arbitration Level (6 lines)	AL	Arbitration Level lines
Grant Acknowledge	GK	Generated by a Master when it has taken control of the Segment
*Proposed New Line	=	-

INTERRUPTS

There are two types of interrupts within the FASTBUS architecture, those which most likely need to cross segment boundaries

and those which usually originate from simple modules and are usually restricted to a single segment. The first type is accommodated in FASTBUS by requiring the interrupting device to perform an extended address block transfer to a specific CSR register. The draft specification gives a recommended interrupt message format.

When a simple Device needs service, it asserts the SR line at any time. The Service Request handler of an interrupt processor, sensing the asserted SR line, can service the simple Device by a variety of methods. In any case, it must arbitrate for and receive Mastership of the bus. Once it is Master of the bus, it can poll the simple Devices to determine which Device is requesting service. Alternatively, it can perform a Broadcast Operation and cause the simple Devices to assert their TP pins, thus causing a bit pattern to appear on the AD lines which corresponds to the Devices that are requesting service.

SEGMENT INTERCONNECTS

The Segment Interconnect (SI), as the name implies, is used to connect two segments together for message passing as required. An SI monitors the activity on the two segments it is connected to, waiting for an address to appear which is in a set of addresses it has been programmed to recognize. It responds to a recognized address asserted on one of the segments (near side) by requesting use of the other segment (far side) and asserting the given address on that segment when it gains control. The two segments remain locked together until the operation is complete. The address asserted on the far side, may, in turn, be recognized by another SI and be passed to yet another segment. An arbitrary number of segmens can be linked as needed for a given operation. The address contains all the information needed to direct the appropriate SI to respond and form the correct connections.

The near-side port of an SI contains a look-up table called a Route Map. The Group Field (GP) of a incoming address is used to determine whether the SI is to pass a incoming message. For every binary combination of GP field the SI must contain three bits of information; "Pass", "Local to Far-Side Segment", and "Base" (lowest GP local to Far-Side). The Base bit is used to accommodate geographical addressing through an SI onto its far-side segment. The local bit is used during a specific type of Broadcast Operation. The Local bit is also used by an SI for diagnostic reasons.

A duplex SI can pass messages in either direction. The specified recommended SI in the draft specification is such an SI. Since messages can arrive at either port of a duplex SI at roughly the same time, a method for resolving these contents is required and has been specified.

An SI sends a non-zero NK, BK response when asserting Address Acknowledge, AK, if for some reason it is unable to pass a message onto its far-side port. A valuable sysem diagnostic feature allows a master to hold onto the bus and remain connected to the SI returning the non-zero NK, BK, response. A master can find out where the network failure occurred and possibly the reason by performing an extended address operation automatically addressing the SI's CSR space. The SI's far and near-side Base and Geographical address, the far side's Bus Halted state, etc. can be read using this technique.

SPARSE DATA SCANS

An important feature of the FASTBUS system is the Sparse Data Scan (SDS) procedure. The SDS procedure simplifies and reduces the cost of

front-end Modules which are typically required in large numbers for certain detectors. Examples of such detectors are drift or proportional wire chamber systems where only a few of the large number of possible data channels actually contain data after a trigger has occured. In these cases, it is most efficient to examine only those channels which contain data and to rapidly skip over those which do not.

A typical FASTBUS SDS system is shown in Figure 14. An SDS controller wishing to find out which SDS modules have data performs a SDS Broadcast Operation. All devices (in that segment) having data assert their TP signal. A module in slot i asserting its TP signal will have that signal received by the SDS controller as a logic 1 on AD; signal line.

After the Broadcast Operation, the SDS controller Geographically addresses all modules which previously asserted their TP signal. Finally, the SDS master reads the data from all SDS controllers.

Since each TP pin is connected to a unique AD line, the SDS master can function without SDS controllers as shown. In such a system, the SDS master would Broadcast to each segment sequentially and then read data only from modules on each segment which asserted their TP signal and thus had data.

MECHANICS AND COOLING

Specifics concerning standard crate and power supply developments and both air and water-cooled crate and rack prototyping work is included in the next section. A major goal of the Fast System Design Group is to achieve module interchangeability between air and water-cooled systems.

HARDWARE AND SOFTWARE EFFORTS SUPPORTING FASTBUS SYSTEM DEVELOPMENT

Several institutions are currently involved in projects intended specifically to support the development of FASTBUS systems as well as the FASTBUS standard itself. The vast majority of the work is being done at the Stanford Linear Accelerator Center (SLAC) and the University of Illinois (U of I). The following summarizes known efforts at all participating institutions at the time of this writing.

Kluge Cards

Fermilab initially developed two kluge cards both having taped interface tranceivers to most all FASTBUS lines. One card permits the use of ECL circuitry only. It has provisions for approximately 180 ICs. The second card translates the ECL bus to TTL via the tranceivers and internally only permits the use of TTL circuitry. It has provisions for 250 ICs. Both cards are wire wrap, two-sided plated through P.C. cards requiring external power bussing.

The U of I and SLAC are presently collaborating on a better general purpose wire wrap, four-layer kluge card which will accomodate ECL, TTL and analog circuits (i.e., ICs requiring ± 15 volts). It will have provision for approximately 350 ICs and will additionally have taping for bus tranceivers and possibly slave Geographical Addressing logic. These cards should be available by late spring and cost roughly \$250.

Both CERN and JWC Corporation in Union, Oregon are also working on kluge cards.

Crates And Backplanes

SLAC is developing the standard crate and backplane. They have contracted out the manufacture of fifty crates and backplanes. All fifty crates have been delivered and as of this writing some twenty-five backplanes have been received.

The crates have twenty-six slots and are louvered to aid in forced-air cooling. The standard card dimensions are in integral Eurocard units.

The backplane consists of nine layers as listed in Table 8. Presently the backplane consists of heavy copper sheets sandwiched together. Production backplanes made by commercial vendors will be true multilayer boards. The fifty pre-production backplanes cost \$1000 each; the crates cost \$350 each. Production backplane costs will be considerably less with a true multilayer board. Production crates should be less than \$200 in large quantities.

The 0 Volt analog pins (0VA) of each slot are each run independently to two central points to prevent ground loops. Slots 0 through 11 0VA pins connect to one point and slots 12 through 25 1VA pins connect to the other point.

SLAC has developed a backplane continuity tester to expedite backplane tests.

An auxiliary space above the standard bus on the backplane provides up to 132 I/O lines at each slot; three columns of 44 pins each with a 0.1 inch grid scheme. This space thus provides tremendous flexibility to users for such things as interconnecting fast signals between modules. Standard flat ribbon connectors can easily be used to interconnect slots as well as interface to external devices.

TABLE 8

FASTBUS BACKPLANE POWER, GROUND AND SIGNAL LAYERS

Layer	Plane	Description
1	Signal	Bus Lines
2	Ground	
3	Power/Signal	+5 Volts,300 Amps Part of Auxiliary Bus
4	Power	-5 Volts,300 Amps -15 Volts
5	Power	-2 Volts,300 Amps +15 Volts
6	Ground	
7	Analog Ground/Signal	0 Volts Analog TP Lines Part of Auxiliary Bus
8	Ground	
9	Signal	Bus Lines

JWC Corporation in Union, Oregon (i.e., Jim White previously from Standard Engineering and Nuclear Enterprises) plans to build a complete single powered and cooled crate system as shown in Figure 15. The system will be roughly 23 inches high and will be powered by standard O.E.M. switching (or linear) power supplies with V,I specifications as required by the user. Total costs for the system excluding the supplies is expected to be less than \$2500 initially. Individual crates will cost \$1400 initially. Fermilab is ordering several such systems.

Module Connector

The standard module connector is a 130 pins 100 grid (0.1 inch) box connector presently available from seven sources. This type of connector is rapidly becoming the industry standard connector. Both Amhdahl and IBM as well as several other large companies use vast amounts of this type of connector. Reliability, low cost and the difficulty of using card-edge connectors on variable thickness multilayer boards led to the FASTBUS committee adapting a box connector. Quite often current requirements on FASTBUS modules will dictate multilayer boards.

SLAC has performed extensive live insertion at full load and insertion and extraction tests on both AMP and Berg connectors with no signs of problems or connector degradation. Bent backplane pins, an early concern of many, has been no problem. A few bent pins occurred while some backplanes (not connected to a crate) were in transit. A few additional tests are planned.

Crate/Module Cooling Tests

SLAC and the University of Illinois have performed several tests on open forced air-cooled and closed air/water heat exchanger-cooled systems. Brookhaven National Laboratory (BNL) has done likewise on a purely water-cooled system. Their work will be reported in another paper given at this conference. CERN has also tested a closed air/water heat exchanger-cooled system. The work done by SLAC and the U of I will be reported here.

The first tests performed were on a forced air single crate system as shown in Figure 16. Hexcell was used to linearlize turbulent air flow from the blower. With all slots filled with "dummy" load modules and the crate power exceeding 2KW, the average die temperature as measured using a transistor base to emitten junction was less than 100° C, a very safe operating point for semiconductors.

The second system tested is shown in Figure 17. The first tests on this closed air/water heat exchanger-cooled system used a Wakefield 9x15x30 inch heat exchanger with 2 kilowatts of power in the loop, 1.5KW from the loaded crate and 0.5KW from the power supply. Temperature sensors were in slots 0,12 and 26. Input water was 16°C at a 2 gallon/minute rate. Sensor temperatures stabilized at 27,34 and 32°C respectively. Water temperature rose 3°C and ambient air temperature rose 15°C.

A second test on the above system used a smaller heat exchanger $(6\times90\times30$ inches). Results were sensor temperatures of 34, 41 and 37°C and a 3°C water temperature rise. Maximum die temperature reached only 93°C.

A closed air/water heat exchanger-cooled system for configurations for multiple crates in shown in Figure 18. The power supplies are mounted either in an adjacent rack or at the rear of the rack.

A system such as this has already been tested at CERN. It differs in that an air/water heat exchanger blower combination unit is at the top of the rack. Cooled air is blown down the sides of the rack. This air enters each crate through plenums on the sides of the crates. Warm air leaves each crate through the top and is drawn to the rear of the crate where it recirculates into the heat exchanger. Warmed air from one crate does not enter the next highest crate. No further details were available on this system at this writing other than successful tests were conducted with the system dissipating a total of nine kilowatts.

Another Water-Cooled System

SLAC and the University of Illinois performed tests on a module enclosed by metal plates to see if convection currents or natural radiation could remove sufficient amounts of heat. The plates were kept cool by fans at room temperature just for purposes of the tests. No external air was allowed to flow through the box. Tests were performed with bare aluminum inner plate walls and with the walls painted black for better heat absorption. Tests also used heat conducting metal to make contact with both the IC and the inner wall of the plate. ICs with heat sinks were also tested for temperature rises.

The conclusion drawn from these tests is that conventional air-cooled P.C. boards can be effectively cooled by simply installing them in a cool-walled box. The IC packages which have less than 700 milliwatts(mw) dissipation can be handled just as they are when being forced air cooled. Packages more than 400-500mw need a heat sink attached. In the cases where a high powered device exists (i.e., >700mw) an extra flexible brass "finger" needs to be attached to make thermal contact with the wall of the box.

The chassis construction for such a system could consist of card slots divided by water-filled plates. These plates could be constructed similar to refrigeration walls. This construction technique results in a crate with fewer than 26 slots. SLAC is now constructing such a crate. Their tests as well as the above tests will be reported at a later date.

Note now the possibilities for total interchangeability between modules in forced air-cooled crates, air/water heat exchanger-cooled crates and water-cooled crates.

Power Supplies and Crate/Power Supply Systems

Much effort in both Europe and the United States is occurring leading to standardized commercially available power supplies and crate/power cooled systems. The JWC Corporation and the SLAC and University of Illinois systems were previously described. Fermilab is designing a switching power supply system as shown in Figure 19. The bin will hold one regulator board, one controller board and up to seven power supply modules. Each power module is easily removable and rests on slides. Power modules can be paralleled. A typical module would provide 80 Amps of current at 5 Volts. 5, 2 and 15 volt modules will be provided. Polarities will be easily reversible. Once designed, commercial manufacturers would build the system.

Todd Products and ACME Corporation, both in New York, are working with both SLAC and the National Bureau of Standards on packaging individual O.E.M. power supplies in a chassis with a front panel for installations where power supplies need not be integrally connected to crates for cooling purposes.

Crate Segment Ancillary Logic/Terminations

SLAC has completed the design of two boards which plug into each end of the backplane (at the rear) as mentioned earlier. One board contains terminations and EG generation (i.e., geographical addressing) logic and the other contains terminations and Broadcast System

Handshake, Bus Halted and Bus Arbitration logic. By May these boards should be available for general use. Cable segment ancillary logic and terminations cards will be worked on afterwards.

Protocol ICs

The University of Illinois has discussed with Motorola the details of having them make LSI protocol and tranceiver ICs using their 68-pin "macro-cell" ECL 10K gate arrays. The new standard JEDEC leadless frame packaging would be used. Specifications of one IC discussed include eight bits of Logical Addressing logic, Geographical Addressing logic, Next Transfer Address logic and input and output latches. The goal is to reduce interface IC count by a factor of 10 to 15 by providing six ICs for interface buffering, slave and master control logic and arbitration logic. Initial design costs would be \$40000 and yield 1-10 pieces. Production costs would be \$47 each in small quantities, \$25 in large quantities. The die size is .95x.95 inch; power dissipation will be between 3 and 4.5 watts. Once the specification is finalized it is hoped that a joint laboratory purchase order will allow development of these ICs.

Segment Interconnect Design/Cable Segment Tests

The segment interconnect conforming to Appendix B of the draft specification has been totally designed (U of I) except for the cable segment drivers and receivers. Preliminary tests by CERN and the U of I have shown that a single-ended ECL system limited to roughly 25 meters in length is practical. Further tests are continuing with a large increase in activity expected in early summer after backplane tests are completed.

I/O Register to FASTBUS Interface and Its Software Support

A simple register to FASTBUS interface initially developed at Los Alamos Scientific Laboratory has been modified and a first prototype taped P.C. board is working at SLAC. Its primary use is for hardware development and diagnostics. It requires 32 bits of output and 32 bits of input from TTL compatible I/O module(s). SLAC has it interfaced to a Digital Equipment Corporation (DEC) LSI-11/2 via commercially available DEC I/O register modules. It is completely program I/O driven and functions with only primitive operations. For example, a register must first be loaded with AD and CB,NH data and then a bit must be set in another register to asset AS, etc. SLAC is now having several units made for distribution to various institutions.

A software package for the I/O Register to FASTBUS Interface written in FORTH is available from SLAC. It currently runs on an LSI 11/2 and uses a special version of FORTH. It is being upgraded to standard FIG-FORTH and versions capable of running on almost any DEC LSI or PDP-11 computer will be available soon. Three levels of "calls" exist. The first is similar to routines which load address and data information and say go; the second, for example, asserts AS and loops waiting for AK, etc.; the third allows users to send primitives such as "assert DS", "release RB", etc.

FORTH, although not structured, is a very compact language with an interactive compiler. It operates stand alone or with an operating system and runs very efficiently timewise with a typical 25% overhead on minicomputers and 85% overhead on microcomputers over assembly language compared to a 1000% overhead for interpreters such as BASIC.

Snoop Module and Its Software

The Snoop module has been designed by SLAC and the University of

Illinois and is in the process of being taped. It is a powerful diagnostic module considered to be extremely important for on-line troubleshooting large systems. The unit consists of a very high speed front end "silo" built as a 100K ECL FIFO, a 68000-based control section and hardware for insertion of WAIT cycles into normal FASTBUS operations taking place on the bus. It additionally has a RS-232 port for terminal communication and a connection to the serial "diagnostic" lines in the FASTBUS segment. The Ethernet protocol is used on this connection.

Software is being written to support its use. Part of the software provides a display so that the Snoop can be used as a FASTBUS logic state analyzer and "glitch" catcher.

The unit and its internal and support software has been described in previous publications 15,16 . Testing of the first prototype Snoop module will begin in late spring.

Miscellaneous Other Support Hardware

A test box (manual segment controller) has been built at SLAC for manual bench testing of modules. It later can be interfaced to a computer for automated testing via an external port.

A passive extender is being developed at SLAC. It will "extend" the bus by requiring insertion in only the left or rightmost slots and removal of one auxillary logic card from its rear connection to the backplane. This card will then plug into the extender card near its front end. No date has been set for this card's availability. There is a need for an active extender card capable of operating in any slot in a backplane.

A LED "display bar" has been developed at SLAC. Its purpose is to give visual conditions of the "DC" state of the FASTBUS. It connects to the bus by means of a buffer card plugged into the rear of the backplane. It mounts at the front top of the crate.

Initial specifications for segment display/diagnostic modules simiar to the Dataway display modules in CAMAC have been written by both CERN and SLAC. FERMILAB is discussing with manufacturers the possibility of their manufacturing a segment display/diagnostic module and an active extender module.

A general test "memory" module has been developed (kluge module only) at SLAC. It has a 256x32 memory and is capable of being read and written in single or block transfer operations to control or data space. The unit is being copied by Fermilab for general system development work.

A fast sequencer developed by SLAC exclusively for testing the FASTBUS at maximum speed is now being tested. It consists of a loadable control store and data memories and, when initiated, executes the preloaded sequence of microwords resulting in FASTBUS operations at very high rates.

System Tests

Phased system tests are being conducted at SLAC to insure the integrity of the system. Phase I tests the bus at fast speeds with the setup shown in Figure 20. These tests will be completed shortly and be reported on afterwards. An additional Phase I test will load a crate with arbitrating modules, some having programmable arbitration levels.

Phase II tests involving multiple crates, Segment Interconnects and cable segments and will be conducted this summer or early fall.

Standard Subroutines for FASTBUS

CERN and Fermilab are using a proposal paper 17 written at Fermilab to develop generally usable FASTBUS software drivers. The intent of the proposed subroutines is similar to that of the "Subroutines for CAMAC" document which attempts to standardize "calls" when writing data acquisition software. A paper will be given on the subject at a May Data Acquisition Conference at Oak Ridge National Laboratory.

System Initializatin and Route Map Generation Software

SLAC is preparing to test system initialization software ¹⁸. This software creates and downloads route maps, tests for improper route map generations, allocates address space and will provide a means for initializing a FASTBUS system, segment by segment. It was written in UCSD (Softec) Pascal Version 2.0. A package capable of running on a LSI 11/23 or 11/2 will be provided.

FASTBUS Diagnostic Language

FASTBUS Diagnostic/Control Language 19 (FDL) is a language definition and interpreter which allow interactive control of FASTBUS systems. The language supports most FASTBUS I/O operations directly. Arbitrary FASTBUS operations can be constructed from primitives. Other language constructs support pattern manipulation, variable length storage allocation, data logging, recursive procedures, and FASTBUS interrupt handling.

FDL is being implemented in PASCAL for LSI-ll's. The proposed standard subroutines for FASTBUS are being used to interface the interpreter to the UNIBUS Processor Interface under development at Fermilab. A driver for the I/O Register to FASTBUS Interface will be written using FDL. Currently the language design is finalized, and the lexical analysis stage of the interpreter is nearly complete. The project is scheduled for completion by this June.

SYSTEMS/INTERFACES

The following summarizes known FASTBUS systems proposed or being developed at the time of this writing. A paper at this conference will describe a system built at Brookhaven National Laboratory using predraft specification protocols, etc. but successfully taking experimental data.

UNIBUS Processor Interface/Data Acquisition System

An interface to FASTBUS from Digital Equipment Corporation's bus, the UNIBUS, is currently under development at FERMILAB. Called the UNIBUS Processor Interface (UPI), it can be broken down into four main parts as shown in Figure 21. A major effort to aid in the development of the UPI is being undertaken at the University of Illinois. They are developing both a language, FSDMAC, for assembling the microcode for the FSDs inside the UPI as well as the operational microcode for the FSDs. Routines are being written in FORTH to debug the UPI.

The UPI consists of one UNIBUS card and two single-width FASTBUS cards. Subsequent developments will be a dual-ported multiple destination bulk memory and a fast data acquisition list-driver processor, the Event Builder. Some or all of these three units can thus provide FERMILAB with data acquisition systems ranging in complexity from that shown in Figure 22 to that shown in Figure 23. The first prototype UPI should be running in late summer. Development of the Bulk Memory and Event Builder should begin soon afterwards.

FERMILAB'S Colliding Detector Data Acquisition System

Colliding detector experiments at Fermilab will begin during the mid-80s. Data acquisition requirements lead to a very large data acquisition system as shown in Figures 24 and 25. 200 to 600 events per second are expected each consisting of 500 to 1000 32-bit words.

The following describes some of the hardware:

 Scanner - This is not a FASTBUS module but possibly resides in a FASTBUS crate.

This module resides in a crate of ADC or TDC modules and does data scanning, sparsification and digitization under control of signals received over an external cable.

2. Controller

This module controls the scanner. It maintains information on all channels it scans in tables of pedestals, thresholds, etc. It contains a two event buffer, and controls up to 1000 front end channels.

Processor

This module formats data (e.g., quadratic corrections and drift times, channel to energy conversion for ADC's), monitors the performance of apparatus, helps perform calibration, etc. It is meant to be a very fast special purpose processor, only moderately programmable. (It also does calculations to be used in later trigger decisions.)

- 4. Interface This is the UPI as previously mentioned.
- Event Builder/Bulk Memory

This will be the high rate node in the data acquisition system. It collects pieces of the event from various controllers and processors and places them into bulk memory for consideration by slower CPU's.

Trigger Processors

These will access the data in the bulk memory and make sophisticated decisions about whether or not to write data to tape.

SLAC/University of Illinois MARK III Detector Electronics

Trigger electronics is being developed at the U of I for use in SLAC's MARK III detector. The electronics is all ECL and consists of 150 modules and five crates. Only FASTBUS mechanics is used. There is no bus interface on the modules. The backplanes are the earlier two layer boards requiring external power wiring. The system is scheduled to run this summer.

Experimental Fast Search and Encode System

The University of Illinois is building a 2500 channel sparse data scan (SDS) system for Fermilab experiment E-401. It will be an addition to an old special hardware system. Each of five crates as shown in Figure 26 will consist of an SDS controller and possibly an SI. The SI's will interconnect the crates to a large dual-ported memory. Each crate will contain up to twenty SDS modules with roughly 25 channels of data each. The memory must take data at a 16 Mbyte/sec rate to match the experiment's data requirements of 1 Mbyte/(one second) spill and 6% dead time. The system is scheduled for completion this fall.

CERN CAMAC Interface

CERN is developing an interface to FASTBUS from and controlled by a CAMAC system. Its structure is sysmetrical in that its FASTBUS and CAMAC interfaces are both similar microcoded logic. Its control circuitry for communications between both microcoded sections is 68000-based firmware. Protocols are message oriented. Hardware implementation is beginning with a tenative completion of early fall.

CERN/UPPSALA MUON COLLABORATION

A pilot FASTBUS project is being developed jointly by CERN and UPPSALA. The first phase will consist of a single FASTBUS crate containing twelve user modules of five different types; an I/O Register to FASTBUS interface, a Romulus read only interface, an SDS controller and a FASTBUS display module (i.e., 2K word history, no microprocessor). The user modules allow the triggering on and recording of data from inelastic muon scatters from 211 scintillation counter hodoscopes.

68000 GENERAL PURPOSE CONTROLLER

SLAC has prepared a preliminary specification for a 68000-based general FASTBUS controller. This slow controller is a device intended for use in monitor and control applications such as monitoring power supplies and temperatures or controlling relays. It will be a complete single board microcomputer with a simple hardware interface to FASTBUS that allows complex FASTBUS operations to be implemented in software. In the 'bare-bones' implementation it will allow the FASTBUS to be driven at speeds limited by the speed of the internal microprocessor. It can optionally be enhanced to drive the FASTBUS in a DMA mode such that FASTBUS utilization will then be limited by processor memory speed.

NIKHEF's CAMAC Transition

NIKHEF in Amsterdam is repackaging into FASTBUS a pair of CAMAC modules, a $68\,000$ controller and a RAM, originally developed for a CERN UAl experiment.

SIN'S Fast Trigger Logic System

A FASTBUS programmable front-end trigger processor system for fast selection of negatively charged particles and for track finding purposes has been proposed for SINDRUM at SIN. Details of the hardware design will become firm this fall; completion is scheduled for the end of 1983. It is currently planned to load the system using the I/O Register to FASTBUS Interface.

TRIUMF's Time Projection Chamber

TRIUMF in British Columbia, Canada has begun development of a fairly large multiple segment FASTBUS data acquisition system for their Time Projection Chamber. When finished it will consist of 7000 front-end channels of ADC data processed down via analog techniques to 2000 channels of octal FASTBUS modules. Data in each crate of ADC modules will be rapidly processed by a FASTBUS bit-sliced/microcontroller and then loaded into a local memory for reading by the host computer. The memory planned to be used is the one developed by SLAC. The host computer will be a Digital Equipment Corporation PDP 11/34. The host interface initially used will be the I/O Register to FASTBUS Interface until the UNIBUS Processor Interface is ready. It is felt that this system requires the full bandwidth that FASTBUS provides. The system will use CERN's rack cooling system. Scheduled completion of this system will be either 1984 or 1985.

BNL MPS II PROPOSAL

A proposal has been written for a system which would apply FASTBUS to the MPS II data acquisition and event selection system for ISABELLE. The system requires 18 crates, 12 SIs, 9 masters of varying requirements and approximately 125 slaves, again with varying requirements.

VAX 11/780 Processor Interface

Both Lawrence Berkeley Laboratory and SLAC are studying feasibility of interfacing Digital Equipment Corporation's (DEC) new 32-bit VAX 11/780 computer to FASTBUS. DEC's general purpose channel interface module, the DR780, would most likely be used for the (32-bit) connection to the VAX 11/780.

OFF-LINE Event Processing System

Fermilab is discussing the possibilities of using 68000, iAPX 432 or VAX like processors perhaps packaged in FASTBUS for a large off-line track reconstruction/event processing system. Additional details will be presented in another paper at this conference.

STANDARD DEVELOPMENT SCHEDULES

An updated working document is presently being prepared for working group consumption only. Efforts to obtain a technical writer to rewrite the final specifications are underway. The past several months have been devoted primarily to prototyping efforts. After these efforts and after all technical features are both clear and entered into the working document final, specification work will begin. This will occur sometime this fall or early winter. The plan is to have the final specification completed by the fall of 1982 for approval at the Nuclear Science Symposium.

ORGANIZATION/PARTICIPATING INSTITUTIONS

The NIM FASTBUS organizational structure is as shown in Figure 27. Participating United States and Canadian institutions are listed in Table 9. NIM FASTBUS efforts are partially funded by the United States Department of Energy.

The ESONE Advanced System Study Group (ASSG) has participated heavily in the development of FASTBUS. Participating institutions are listed in Table 10.

TABLE 9 NIM PARTICIPATING INSTITUTIONS

Alberta, University of
Brookhaven National Laboratory
Fermilab
Lawrence Berkeley Laboratory
Los Alamos Scientific Laboratory
Massachusetts Institute of Technology
Michigan, University of
National Bureau of Standards
Oak Ridge National Laboratory
Saskatchewan, University of
Stanford Linear Accelerator Laboratory
Dick A. Mack (Consultant)

TABLE 10 ESONE ASSG PARTICIPATING INSTITUTIONS

CEA, Saclay CERN, Geneva DESY, Hamburg IHEP, Serpukhov IKO, Amsterdam INFN, Geneva INFN, Rome LAL, Orsay LNPI, Gatchina NIKHEF, Amsterdam RISO, Rosklde SGAE, Vienna SIN, Villigen Technical University, Vienna ULCC, London (and others)

ACKNOWLEDGEMENTS

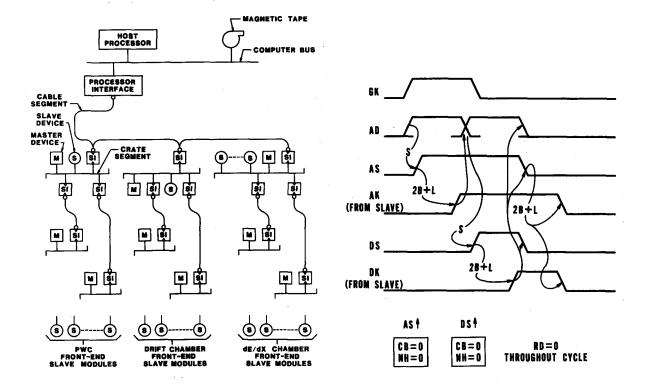
I want to personally thank all the people who aided me directly or indirectly, in preparing this paper. The number of people is very large; their cooperation and help richly appreciated. An attempt was made to name institutions whenever discussing specific work in order that FASTBUS representatives from these institutions can be contacted for further information or direct interested people to proper personnel.

REFERENCES

- "Future Data Bus Requirements for Laboratory High Speed Data Acquisition Systems", U.S. NIM Committee U.S. ERDA TID 27621, June 1977.
- "FASTBUS Notes for Draft Description", October 1978. 2.
- 3. "FASTBUS Interim Draft for Comment", May 1979.
- "FASTBUS Draft Tentative Specification", January 19.
 "FASTBUS Draft Tentative Specification", March 1980.
 "FASTBUS Tentative Specification", July 1980. 4.
- 5.
- 6.
- "Addendum to FASTBUS Tentative Specification dated July 1980", September 1980.
- "FASTBUS Tentative Specification", September 1980.
- "Addendum to FASTBUS Tentative Specification dated September 9. 1980,"
- November 1980.
 R. S. Larsen, "Status of the FASTBUS Standard Data Bus", 10. presented at the Nuclear Science Symposium, November 1980.
- D. B. Gustavson, "FASTBUS Software Status", presented at the Nuclear
- Science Symposium, November 1980. B. Wadsworth, "A System Overview of FASTBUS", IEEE Transactions on
- Nuclear Science $\underline{\text{NS-27}}$, No. 1 (1980), pp. 612-621. 13. Amp,Bendix,Berg,Eby,Elco,ITT Cannon, Philips
- 14. Larry Leipamer, title unknown at this writing, presented at this conference.
- 15. H. Walz and R. Downing, "FASTBUS SNOOP DIAGNOSTIC
- MODULE", Nuclear Science Symposium, Nov. 1980.

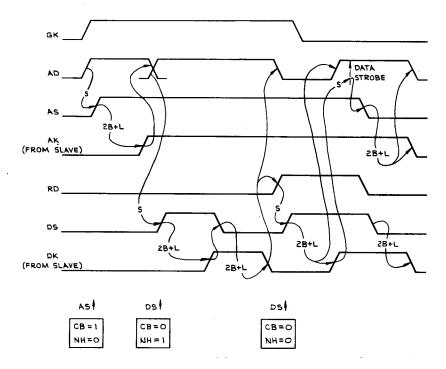
 16. D. B. Gustavson et al ., "A 'Front Panel' Human Interface for FASTBUS." Nuclear Science Symposium, Nov. 1980.

- S. E. Gannon and L. M. Taff, "Proposal for Standard Subroutines for FASTBUS", FSDG-085, July 1980.
 S. Deiss, "FASTBUS System Manager Demonstration", FSDG-097, Nov 1980.
 K. D. Nater, "FASTBUS Diagnostic/Control Language", FSDG-092, Oct 1980.
 T. Nash, "A Review of Programmable Systems Associated With Fermilab Experiments", presented at this conference at this conference.

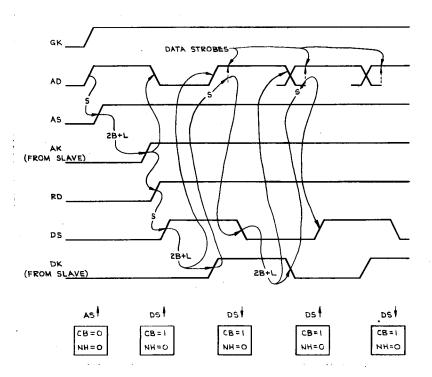


Large Detector Data Acquisition System Implemented With Fastbus Figure 1

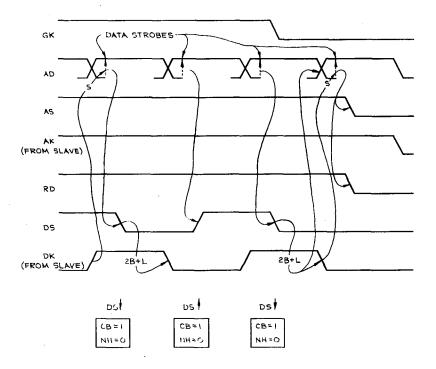
Timing (As Seen At Master) For Single Work Write To A Specific Address In Data Space. Figure 2



Timing (As Seen At Master) For Three Cycle Extended Address Single Word Read From A Specific Address In Control Space
Figure 3

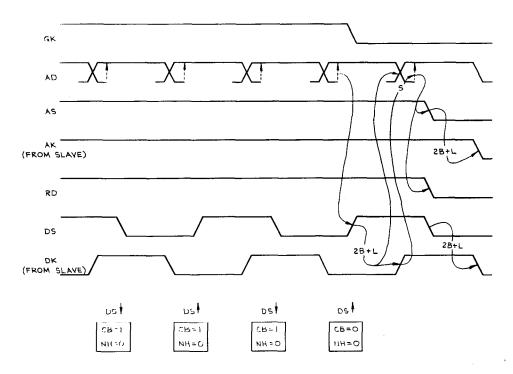


Timing (As Seen From Master) For Block Transfer Read From A Specific Address In Data Space Figure 4

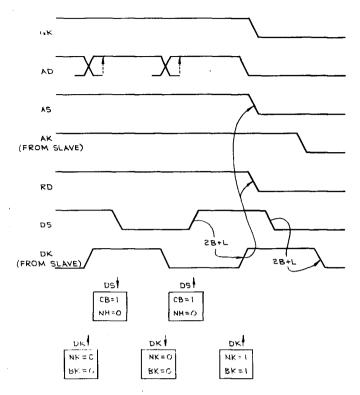


Timing (As Seen By Master) For Block Transfer (Read) Termination By Master Even Number Of Words Transferred

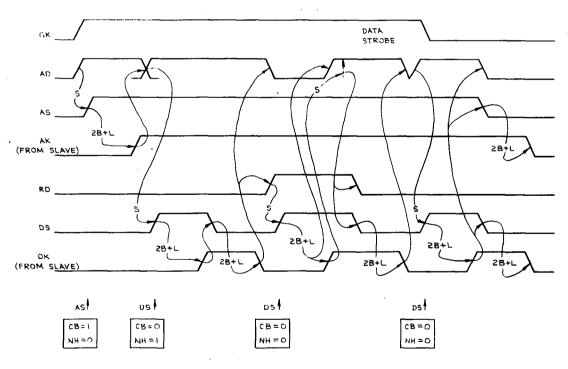
Figure 5



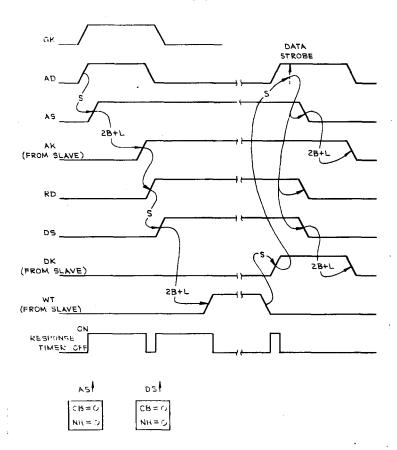
Timing (As Seen By Master) For Block Transfer (Read)
Termination By Master-Odd Number Of Words Transferred
Figure 6



Timing (As Seen By Master) For Block Transfer
Termination By Slave-Receipt Of Non-Zero
NK, BK Response From Slave
Figure 7

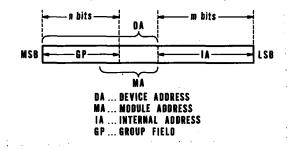


Timing (As Seen By Master) For Mixed Read-Modify-Write Operation Extended Address Single Word Read-Modify-Write From/To A Specific Address In Control Space
Figure 8

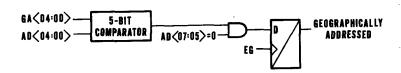


Timing (As Seen By Master) For Single Word Read From A Specific Address In Data Space-Slave Response Delayed By Use Of Wait (WT) Line

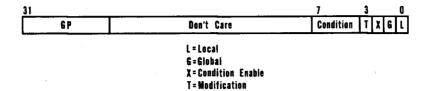
Figure 9



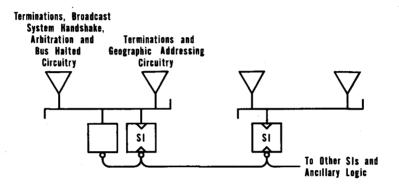
Address Fields Figure 10



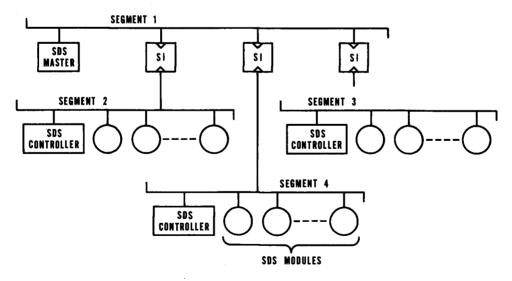
Typical Geographical Addressing Circuitry
Figure 11



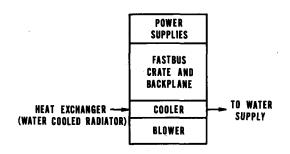
Broadcast Address Fields Figure 12



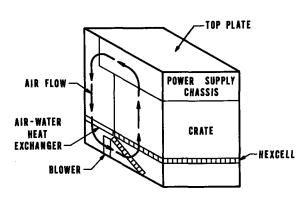
Physical Implementation Details Ancillary Logic Figure 13



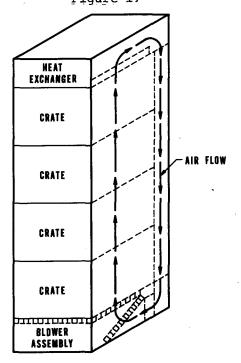
A Sparse Data Scan Implementation Figure 14



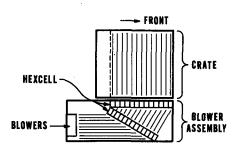
Proposed JWC Corporation Powered Cooled Crate System Figure 15



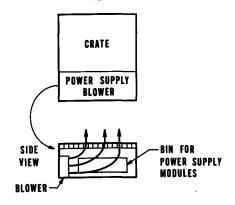
Single Crate
Closed Air/Water Heat Exchanger
Cooling
Figure 17



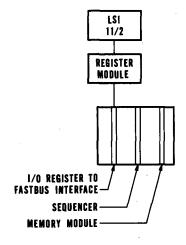
Multiple Crates Closed Air/Water Heat Exchanger Cooling (Figure 18)



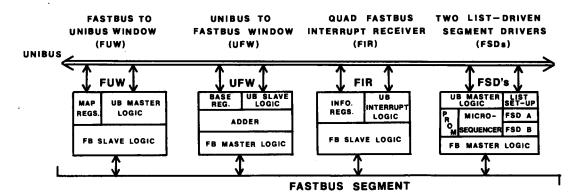
Single Crate Forced Air Cooling (Side View) Figure 16



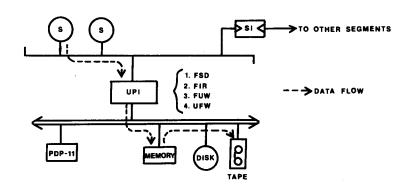
Proposed Fermilab Crate/Power Supply System Figure 19



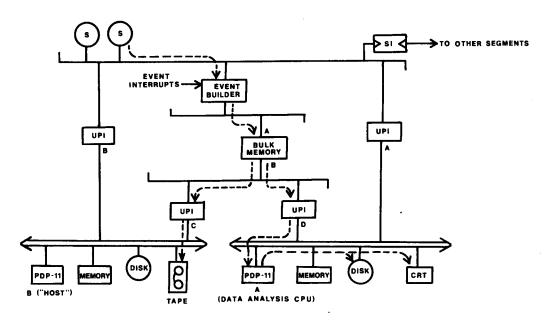
Phase I System Test Setup Figure 20



Unibus Processor Interface Functional Block Diagram Figure 21



Simple UPI Data Acquisition System Figure 22



Fermilab "Future" Data Acquisition System Figure 23

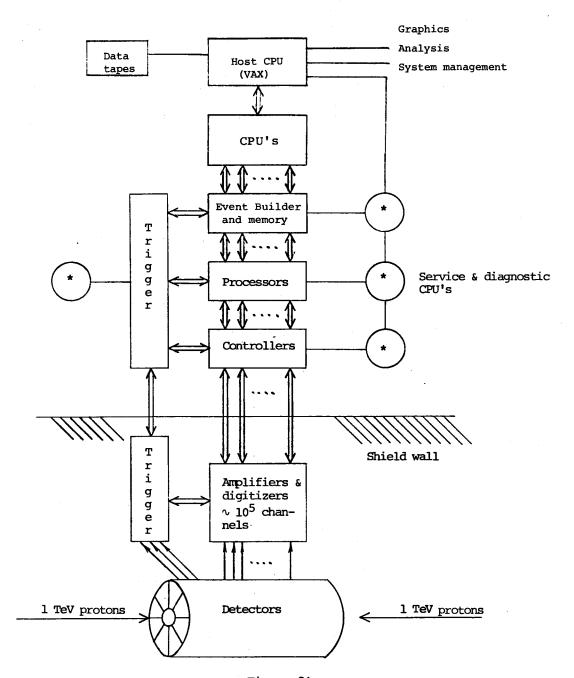


Figure 24 Colliding Detector Data Acquisition System General Layout

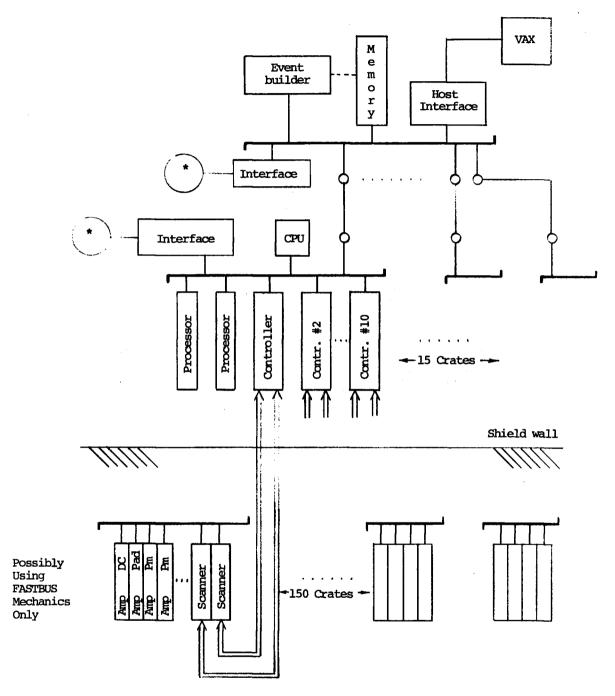
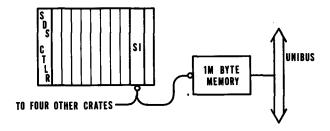
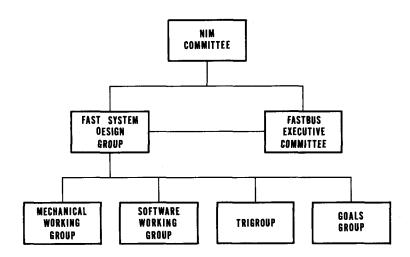


Figure 25
Colliding Detector Data Acquisition System
FASTBUS Implementation



Sparse Data Scan Search And Encode System Figure 26



NIM Fastbus Organization Figure 27