# FASTBUS STATUS FROM A SYSTEM DESIGNER'S POINT OF VIEW*

SLAC-PUB-2753
June 1981
(I)

David B. Gustavson
Computation Research Group
Stanford Linear Accelerator Center

## Abstract

Designers of modern systems for data acquisition, control and data analysis must provide the needed throughput, response and usability while avoiding economic and complexity limitations. Today's technology makes multiple-processor systems economically attractive, but such systems tend to be complex. In addition, rapidly changing requirements mandate easy reconfigurability and expandability.

Though named for its potential speed, FASTBUS will make its greatest contribution to most systems with its flexibility and versatility. FASTBUS provides a framework in which the designer can build systems ranging from small and simple to very large and powerful in a flexible way, always leaving room for future expansion and allowing for simple tailoring of the system as needs change.

This paper examines the features of FASTBUS relevant to the design of such systems, and discusses the current status. Though FASTBUS will be cost-effective even for most small systems, this paper will emphasize the large-system aspects.

## Introduction

FASTBUS designers have given serious consideration to the programming, communication, debugging and diagnostic problems inherent in the implementation of a general multiple-processor multiple-bus system, and to the need for flexibility and expandability of systems.

Management tools have been included for coping with large systems. These range from sophisticated system-initialization and management software to simple standardization of control and status features in an attempt to reduce system complexity. Easy reconfigurability and expandability have been provided by the large address space and a flexible bus segment interconnection scheme which allows adding new data pathways wherever bypasses are needed for heavy traffic, without topological restrictions.

Sophisticated diagnostic capabilities have been included in addition to the essential multiprocessor communication features. These include the Snoop module, which is an intelligent logic analyzer and human interface for the FASTBUS system; the Serial Diagnostic Network, which interconnects Snoops and other diagnostic tools; and error detection features including parity checking and route tracing through the Segment Interconnect modules which connect the separate buses in a system.

## System Management Facilities

The FASTBUS standard[1] provides several features which aid in the management of large systems. A large 132-pin auxiliary connector area is included at every module position in the backplane of the FASTBUS crate, which for many applications is sufficient to carry all the user's signals to or from the module. This allows cables to be semi-permanently attached to the back of the backplane, reducing the chance of incorrectly recabling after replacing a defective module.

A position-dependent form of addressing, called Geographical Addressing, is provided to allow accessing whatever module is in a given physical location. Every module contains an identifying register which can be read to determine the module's type or model. Accessing this register via Geographical Addressing allows automatic procedures to verify that the correct module is in place at each physical position. Combined with the backplane-mounted cabling mentioned above, this increases the probability that the correct type of module is processing a given set of signals.

Control and Status Registers are defined in a standard way within each module so that one need only know how to address the module itself, knowing nothing else about the module's properties, in order to be able to access any implemented register in the module. Any FASTBUS addressing mode may be used: geographical, logical or broadcast. The identifying register mentioned above is an example of a Status Register. Control registers may be used to specify operating parameters for a module, such as its arbitration level (priority) or logical address. The access mechanism has been specifically designed to make it possible to broadcast commands or data to control registers in several modules at once.

A Logical Addressing mechanism is provided so that a module can use however much address space it needs, independent of its physical size or position on a segment. Allocating logical addresses to modules is potentially an error-prone process, because the address range used by one module must never overlap that used by any other. Furthermore, addresses of modules on each bus segment must be grouped together so that the high-order bits can be used to discriminate between segments, and the quantum of addresses which can be allocated to individual segments varies in size depending on the capabilities of the Segment Interconnects on those segments.

Since modules which use logical addressing have their addresses set by a writeable control register, the allocating and setting of logical addresses can be automated.

Modules which can become bus masters must have an arbitration level, or priority, assigned which is unique for each module on a segment. Errors in setting priority are very serious, as they can result in two modules gaining mastership of the bus at once. Since the priority is contained in a writeable register, however, this process also can be automated. The normal priorities (levels 1-31) remain local to each segment, and therefore can easily be checked segment by segment.

System priorities (levels 32-63), however, propagate from the master along the connecting path through the system until they reach the destination segment containing the slave, and may also be used in broadcasts. These levels must generally be assigned uniquely in a system, but under certain conditions the same level may be assigned to several devices. This may happen, for example, if the

* Work supported by the Department of Energy under contract number DE-AC03-76SF00515.

(Contributed to the Topical Conference on Computerized Data Acquisition in Particle and Nuclear Physics, Oak Ridge, Tennessee, May 28-30, 1981.)

same level is assigned to the outgoing side of several Segment Interconnects leaving the same segment or suitably restricted set of segments. These assignments are especially appropriate for automatic checking.

Segment Interconnect modules must have tables of address ranges which they are to pass, one table for each direction. Address assignments to these tables must follow certain rules in order to guarantee unique connection paths between master and slave. Here again, the system has been designed to permit automation of the table generation, loading and checking process.

Another problem which is significant in large systems is the record keeping associated with each module, keeping track of revision levels, documentation, calibration information, repair records, ownership etc. FASTBUS allocates a large block of status register addresses for these purposes, with the information organized in a way which allows easy implementation using 8-bit-wide PROMs, EPROMs, or EEROMs. There is plenty of room for storage of these records and even perhaps a small instruction manual in one or two chips within the module itself. This optional information, along with the module type identification which is always available, makes automatic generation of equipment logs possible. If fully implemented, these features would allow easy display of a list of modules which have been repaired or replaced recently when problems are discovered after shutdown periods or shift changes.

A System Manager program[2] has been written in a widely available version of Pascal (UCSD Pascal). It provides convenient facilities for entering a description of the desired system, verifying that the real system corresponds, and performing the address assignments and checks described above, as well as generating the Segment Interconnect route table entries. It even allows manual intervention in the assignment process, checking the requested special assignments for feasibility and correctness. It then can use the Geographical Addressing mechanism to access and initialize a system with the generated or assigned values.

This program has been tested in simulated systems, and now awaits real hardware before final implementation of the real initialization system.

## Multiprocessor Communication Facilities

FASTBUS provides a framework for multiprocessor systems. Each Segment is a shareable bus, with an arbitration mechanism to determine which of many potential competing masters (processors) will gain control at any one time. Since any bus will become a bottleneck if too many processors use it at once, FASTBUS provides for multiple bus segments which operate independently in order to increase the system bandwidth. Most processors will need their own internal working memory so that they do not require continuous access to their Segment, which would interfere with other users.

### Arbitration

The FASTBUS arbitration mechanism assigns each potential master of a segment an arbitration level or priority which is unique on that segment. A problem of bus hogging can arise if a high-priority master demands continual access to the bus, freezing out other users. The word "priority" is a bit misleading in FASTBUS, however, since there is no mechanism which allows high priority modules to preempt bus use by lower priority modules. Usually bus access is likely to be first come, first served, at least in systems which are not overloaded. The real reason for arbitration levels is merely to prevent two masters from using the bus simultaneously, garbling one another's signals.

In order to allow "fair" access to the bus, a new mechanism has recently been added, now called the "assured access protocol." Modules may ignore this protocol and be bus hogs if the system implementer allows it, as this seems logically necessary in some applications. However, most modules in most applications will be set to obey the protocol, which does not allow one module to have a second access to the segment until every waiting module has had a turn. This tends to reduce the "priority" aspects of the arbitration level assignments to a negligible value.

For systems with special requirements, dynamic arbitration level assignment changes are possible, because the levels are specified in writeable control registers. A master with high "priority" can seize the segment, rewrite every other master's arbitration level, and then release the segment.

### Addressing

Once a processor has become master of the segment, it can communicate with any other module in the system by asserting its address. FASTBUS has a very large address space, which is divided (unevenly) among all the segments in the system. Most modules reside on backplane segments, which provide mechanical support, power, cooling, etc. Cable segments, which are long and flexible, provide only the logical bus signals and returns, and are primarily used for interconnecting backplane segments, though any FASTBUS device could be designed to connect directly to a cable segment.

Since address assignments can change as the system grows, processors will most conveniently refer symbolically to the other FASTBUS devices they access. Values can be assigned to these symbolic references by a linking process similar to that used presently in high-level languages in single-processor systems. The system manager program produces a symbol table which provides the current address assignments needed in the linking process. A variety of other methods can also be used for linking. For example, processors can request current address information from the host computer, which maintains the system description data files.

When an address appears on a segment, Segment Interconnect (SI) modules look it up in their tables in order to decide whether they are responsible for making a connection to another segment in order to pass the address along. If so, they stand in place of the master on the next segment, waiting for mastership, etc., then assert the address again on that segment. Other SI's may recognize and connect in turn until the operation has propagated its address to the segment containing the addressed module.

Thus the entire system is united by sharing one address space, and any master can access any other module in the system merely by asserting its address, yet the various parts of the system work independently at times when interconnection is not needed.

Of course, interconnection is costly because it reduces the parallelism in the system, forcing all segments linked by the connection to act together as one. If connection is needed frequently between two segments, care should be taken to minimize the number of segments used to link them. The FASTBUS interconnect scheme makes it easy to add a cable segment between two points whenever it is needed, because there is no topological restriction. (Some interconnection techniques would require the system to be tree-like, with no cross connections between branches, or no loops.)

## Bus Timing Adjustments

Another problem handled by the FASTBUS design is the adjustment of timing parameters which depend on bus properties. When operations connect through long cables or multiple segments, the propagation time for signals and responses increases, and the nonuniformity of the cables used will cause some signals to travel faster than others (signal skew). In the FASTBUS scheme, each master needs to know only the skew properties for the segment to which it is attached, and each slave needs to know only its own internal timing requirements. All changes in timing which are caused by the particular path or particular modules used in the connection are taken care of by the Segment Interconnect modules, and by the handshakes built into the FASTBUS protocol.

## Resource Management

In multiprocessor systems it is frequently necessary for several processors to coordinate their activities so as to allocate nonshareable resources. For example, generally one wants only a single processor to write on a given magnetic tape at a given time.

In order to manage resources of this sort, generally some form of shared data must be used, such as a table stored in commonly accessible memory. Mechanisms must be provided to allow one processor to update such a table completely in an internally consistent way before another processor uses the table or modifies it. These mechanisms, called mutual exclusion mechanisms, require at the minimum an uninterruptable test-and-set operation which can operate on a flag entry in the shared memory.

Though test-and-set operations are logically sufficient for any application, they would be rather inefficient by themselves because the details of the allocation of every shared resource (which may include simple things like signal paths) would have to be taken care of in software. Therefore, more powerful mechanisms have been included in FASTBUS, which allow many kinds of resources to be managed by the hardware without direct software intervention.

There are two mechanisms for mutual exclusion built into the FASTBUS protocols, and a third implemented in the Control and Status Register system. The most primitive is a generalization of the read-modify-write, in which a master connects to a single slave and excludes other masters from interfering with their communication by means of the Address Strobe/Address Acknowledge (AS/AK) lock. The master may lead the slave through a read-modify-write or a more complex operation, possibly even including a block transfer, without danger of interruption.

The more complex and more general mechanism involves the master's control of the bus arbitration process. If the master does not release the bus (by removing its GK assertion), no arbitration can occur and thus no other master can gain mastership of the bus in order to interfere. This permits one master to safely perform arbitrarily complex operations with multiple slaves, one at a time or simultaneously by using a broadcast.

In addition, the Segment Interconnect devices have been designed so that once they establish a connection between segments, they will not release that connection until the connecting master gives up its mastership of the bus. Thus, even though subsequent operations involve addresses which do not use that SI's connection, the connection is maintained so that no masters on the connected segments can interfere with any slave which has been addressed at any time by the current master. This allows a master to perform protected sequences of operations on several slaves even though the slaves do not reside on the same segments. A master can perform address-only operations to each slave in order to make sure all the needed connections are possible and that all the slaves exist before performing irreversible operations on any of them.

The third mechanism, built into the Control and Status Register system, implements a form of test-and-set in the module hardware. A module which controls a particular resource, such as a tape drive controller, contains a special resource-controlling register into which the master tries to write his own identifier. If the module is not in use at the time, the write succeeds. If the module is in use, the write fails and the master is notified. When the current user is finished with the module, it resets the status of the module to "available," reenabling write access to the resource-controlling register so that another processor may gain access.

## Other Communication Links

There are other ways for processors to communicate in a FASTBUS system. For example, they may use the Serial Diagnostic Network, originally proposed as a solution to the maintenance and diagnostic problems in a multisegment system, but now recognized as a more generally useful resource. This network will be slow (hundreds of kilobits per second) in the early FASTBUS systems, but will be upgraded to the 10 megabit IEEE-802 standard when suitable interface chips become available.

Other point-to-point links are envisioned for FASTBUS as well. These share with the Serial Diagnostic Network the problem that no direct handshake connection between sender and receiver exists during the transfer, so higher-level protocols are needed. These protocols generally involve the return of acknowledging messages to the originating processor. The connections made by the Segment Interconnect modules, on the other hand, always result in a cycle-by-cycle handshake and status response from the slave (except in the unusual case of nonhandshake block transfers, where the response is delayed by the system propagation delay while the data transfer is proceeding in a pipelined mode).

# Diagnostic Capabilities

Diagnostic capability has been a matter of fundamental concern in the design of FASTBUS. Experience has shown that isolating faults and detecting intermittent failures is difficult enough in single-processor systems, and without good tools the problems could prove insurmountable in multiprocessor systems.

The bus protocol itself includes diagnostic features. It has been designed so that any operation can be examined and single-cycle-stepped by diagnostic equipment, allowing slow equipment (such as microprocessors or even humans) to examine every part of every operation, monitoring internal state changes within modules, etc.

## Parity

Parity has been included on the bus in order to detect intermittent transmission failures. Though parity is optional, in order to accommodate the very large numbers of very low cost data acquisition modules envisioned for some applications, most modules of any complexity should include it.

Many modules will contain registers which can be read and written as part of a background task which works its way around the system checking modules and pathways. The Next Transfer Address register found in many modules

may be used for this purpose without disturbing any internal module functions, and the identifying register found in all modules can be read.

Postmortem Information

The FASTBUS specification requires that almost all information contained in Control or Status Registers be left undisturbed by a hardware Reset Bus command which is likely to follow a system crash or disaster of some sort. All modules are disabled, and a few control bits are reset, but most of the module's pre-crash parameters are available for diagnostic purposes. For similar reasons, write-only registers are generally prohibited.

A special feature of the Segment Interconnect module is a Route Trace capability. If system address-routing problems are encountered, a special broadcast command can be followed by a sequence of probing operations which reveal the exact path taken by the operation which failed.

Snoop Module

A special diagnostic module called the Snoop[3,4] has been designed in parallel with the FASTBUS development in order to ensure that the bus has the features necessary to support the diagnostic system. The Snoop is a special purpose logic analyzer which maintains a history of bus signals for the last 250 cycles, has trapping hardware which can stop bus operations to certain addresses containing certain data patterns, allows single-cycle-stepping of the operations, has facilities for taking 250-word snapshots of bus activity and can concatenate snapshots to form a continuous record when necessary.

The Snoop also contains hardware to act as a master or a slave, and a microprocessor which coordinates the parts and provides a comfortable human interface. Because some system problems will require coordinating information from several segments at once, Snoops communicate with one another over the FASTBUS Serial Diagnostic Network. This allows communication even when the FASTBUS itself is busy or broken. The human interface is via a smart terminal which connects to one of the Snoops and can connect to any Snoop through the network. The terminal also provides storage on diskettes for test programs and data.

Information about address assignments and the system structure is centralized in files maintained by the System Manager program mentioned above, and can be accessed via the network as needed for diagnostic work.

Since most parameters of modules are specified by writeable registers rather than the more familiar switches, the Snoop provides a human interface to the module parameters by displaying the register contents on a screen and allowing changes to be made as though one were editing text. Snapshots of module status can also be taken and displayed while the system is in operation. Such manual intervention in a module's operation is dangerous in running systems, but is an invaluable capability in diagnostic situations, on the development or test bench, or in small systems where the Snoop itself may act as the initializing Host or primary data acquisition computer.

Many of these capabilities are also available in a lower-speed implementation by simulation, using a simple register interface to the bus with a few high-speed circuits which act to slow the bus to the speed of the simulating computer[5].

## Future Expandability

One of the main properties of FASTBUS which facilitates future expandability is its large address space, which provides directly addressable access to $2^{32}$ 32-bit words in a connected system. Even this limit can be exceeded, should that ever become necessary, by using an extended protocol as discussed below. The fundamental limit is then about $2^{29}$ addressable devices, each with any desired amount of internal address space. The fundamental limit on the number of independent segments is $2^{24}$. In practice, $2^{32}$ words should be an adequate address space for quite some time.

The address extension mentioned above is similar to the addressing mechanism used for accessing Control and Status Registers, which allows $2^{32}$ such registers per module. This is not likely to be a limiting factor.

FASTBUS flexibility extends beyond the reconfigurability discussed in previous sections. Because of the nature of its protocols, new generations of modules may be developed which multiplex more information in each operation. FASTBUS protocols are driven by the master. There is no control sequence information which appears at the beginning of an operation and governs the rest of the operation's cycles as is the case for many other buses. Thus the bus itself, the Segment Interconnect modules, and the diagnostic facilities are independent of the cycle sequence and contents after the address cycle has connected the slave to the master.

This design approach allows new systems to be created which pass additional information by additional levels of multiplexing in each operation. For example, access control passwords, data descriptors, capability qualifiers or wider data fields could be added to the basic protocol in a way which would not interfere with the operation of other modules, and the new system could still use the same Segment Interconnects, etc.

In fact, once the AS/AK lock has been established, the communicating master and slave can use their own private protocols, redefining most bus lines as they wish, without disturbing any other modules.

Additional expandability can also be achieved through the network connection. By using an industry standard network protocol, FASTBUS achieves easy access to a wide variety of sophisticated peripheral devices, particularly displays, disk memories, and high-quality printers.

## Availability

FASTBUS is now well into the hardware prototyping phase[6]. Fifty backplanes and crate assemblies have now been built and distributed to cooperating laboratories. Prototype test modules are now in operation. The Snoop hardware design is complete and circuit board layout is nearly done. Wire-wrapped versions of the microprocessor portion are in progress. Software development is well along. The Segment Interconnect is designed except for some details about the cable segment connection. Initially cable segments will probably be limited in length due to the need for noise immunity on certain signal lines.

Several projects are underway to produce processor interfaces to FASTBUS. The next year should begin to see FASTBUS pieces produced by industry, though most new designs will still be done by the laboratories. The serial diagnostic network has been implemented and tested in a low-speed but compact version for prototype work. The IEEE-802 standard is expected to appear in draft in June, 1981, with the Carrier Sense Multiple Access portion very similar to the Ethernet standard proposed by Xerox, DEC and Intel[7]. Chip sets to support this standard should be available in about a year.

A single-segment system using a protocol similar to FASTBUS (an early version, modified for prototype purposes) has been in actual use in a data acquisition system at Brookhaven National Laboratory[8].

## Summary

FASTBUS will soon be available for use as a framework in which to build multiprocessor data acquisition, analysis and control systems. Though large and complex systems will continue to have their problems, FASTBUS will prove to be more a part of the solution than a part of the problem. Since FASTBUS will also be cost-effective for small systems, the growing pains always encountered as applications develop beyond their original conceptions should be much reduced.

## References


1. FASTBUS Draft Specification, September 1980, U. S. NIM Committee, available from L. Costrell, Department of Commerce, National Bureau of Standards, Washington, DC 20234.

2. Available from S. R. Deiss of the Stanford Linear Accelerator Center.

3. D. B. Gustavson et al., "A 'Front Panel' Human Interface for FASTBUS," IEEE Transactions on Nuclear Science, Vol. NS-28, No. 1, February 1981, pp. 343-345.

4. H. V. Walz and R. Downing, "FASTBUS Snoop Diagnostic Module," IEEE Transactions on Nuclear Science, Vol. NS-28, No. 1, February 1981, pp. 380-384.

5. C. A. Logg et al., "FASTBUS Introduction and Demonstration," submitted for publication in this issue.

6. R. S. Larsen, "Status of the FASTBUS Standard Data Bus," IEEE Transactions on Nuclear Science, Vol. NS-28, No. 1, February 1981, pp. 322-329.

7. "The Ethernet, A Local Area Network: Data Link Layer and Physical Layer Specifications," Version 1.0, September 30, 1980, available from Intel Corporation, 3065 Bowers Avenue, Santa Clara, CA 95051.

8. L. B. Leipuner et al., "A FASTBUS System used in a High Energy Experiment," IEEE Transactions on Nuclear Science, Vol. NS-28, No. 1, February 1981, pp 333-335.


See also companion papers on FASTBUS by L. Paffrath and R. Downing submitted for publication in this issue.