SLAC-PUB-2378
August 1979
(I)

AN INTRODUCTION TO THE FASTBUS

MASTER

David B. GUSTAVSON

Stanford Linear Accelerator Center*
P. O. Box 4349, bin 88
Stanford, CA 94305

The Fastbus is a modular data bus system for data acquisition and data processing. 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 or short paths, using handshake protocols for reliability. It can also operate synchronously without handshakes for transfer of data blocks at maximum speed. The goals, history and motivation for the Fastbus are summarized briefly. The structure of the Fastbus system is described in general and some details of its operation are introduced.

HISTORY AND GOALS OF THE FASTBUS DESIGN

In 1975 several physicists active in high energy elementary particle physics asked the NIM (Nuclear Instrumentation Modules) committee to consider development of a standard data acquisition system which would meet the needs of future physics experiments and which would be of general utility. The NIM committee established a study group to examine the needs and determine objectives to be met by such a system. In 1977, a design committee was formed to begin the actual design of a system which would meet these goals. The design and prototyping efforts have been supported by the U. S. Department of Energy.

The design committee has not yet completed its work, but the fundamental Fastbus architecture is fairly firm now and work is proceeding on the finer details. A prototype mechanical package has been designed and is being produced in small quantities for use in prototyping activities at various laboratories. The descriptions which follow reflect the current thinking of the design committee. Some details may be changed before the standard is final. However, details at the level described in this paper are not likely to change much.

The goals of the design are:

Highest possible speed, in order to handle the high data rates encountered in complex particle detectors, and reduce the temptation to design custom systems for each application.

Lowest possible cost, because huge volumes of electronics are required and any unnecessary cost is multiplied by a large factor.

Sparse-data scanning capability, because the relevant data is usually a tiny fraction of the data available (most elements of a large detector are not hit by any particles at all in a typical event).

Modular construction, so that useful standard building blocks can be developed which can be connected in various ways to meet the needs of various experiments.

Multiple processor organization, so that many low cost computing elements can

* Work supported by the Department of Energy under contract DE-AC03-76SF00515. (Presented at the Conference on High Energy Physics & Nuclear Structure, Univ. of British Columbia, Vancouver, B. C., Canada, August 13-17, 1979.)

be brought to bear on parts of the data acquisition and analysis problem.

Segmented organization, so that the multiple processors will not be overly limited by the bus bandwidth. Segments of the bus can operate independently, joining together automatically as needed for passing data.

Asynchronous operation, so that slow and fast modules can be mixed in the system, and so that the system speed can increase easily as technology advances.

Synchronous operation, so that data transfer can occur at the full bandwidth capability of the transfer medium when ultimate performance is needed.

Uniform protocols, so that no protocol translation is needed as data flows from segment to segment. The Fastbus uses cable segments to join segment interconnect modules which may reside on backplane segments, rather than a branch highway cable connecting crate controllers as in CAMAC.

Uniform addressing, no dedicated positions on backplanes. Modules have an address which is unique in a connected system and which is independent of module position within a segment. All module positions are equal in capability.

These goals, though partially conflicting, have been met to a remarkable degree by the Fastbus design.

HARDWARE AND ORGANIZATION OF THE FASTBUS SYSTEM

The essential part of the Fastbus design is the bus protocol. Various electrical and mechanical implementations are possible which use this protocol and hence are easy to interface to one another. However, one implementation has been chosen for prototype work at SLAC and several other laboratories, so that our first module designs will be both physically and electrically compatible. This implementation appears to be satisfactory for most applications, and will doubtless evolve into a standard. Other options, including conductive cooling, on-card power conversion and various bus signal levels, are also being considered and prototyped.

This implementation consists of a crate which fits in a standard 19-inch rack, with a multilayer backplane at the rear and card guides top and bottom. Cooling air flows vertically from bottom to top through the crate, driven by separate fan units which may serve several stacked crates. Water cooled rechillers can be mounted between crates as necessary. The design capacity of 1500 watts per crate seems easily accommodated. Doubling this may be possible.

A crate holds 26 modules, which consist of circuit boards of approximately 323 mm by 340 mm. This size fits within the "EUROCARD" sizing formula. The rear edge of the boards contains two box connectors which mate with square pins which protrude from the backplane. The main bus connector has two rows of 65 positions on 0.1 inch centers. The optional auxiliary connector has up to three rows of 44 positions, and connects to long non-bussed pins which protrude from both sides of the backplane so that user cabling can be attached to the backplane instead of to the module.

The main connector supplies power and ground as well as signals, with one ground to every four signals. Power is distributed by heavy layers of copper in the backplane, providing for 300 amperes at 5 and -5.2 volts, 150 amperes at -2 volts, 50 at +-15 volts, and 50 amperes each for two auxiliary supplies. A quiet analog ground line is also provided. Power supplies are separate from crates, connected by remote sensing cables. Typically they will be mounted in the rear of the rack. The 50 ampere buses could provide the power for on-board converters.

Bus terminators and a small amount of logic associated with bus arbitration will

reside on small boards mated to extended main connector pins on the back of the backplane in the end positions. All bus signals are emitter-coupled-logic levels.

SEGMENTATION OF THE FASTBUS

The backplane bus in a single crate is a single segment, which means that the bus can be driven by only one device at a time. Thus, although multiple processors may be plugged into a single crate and share a single backplane bus, only one of them may use it at a time. Contention for the use of the bus may reduce the throughput of the system by causing processors to wait.

Distributing the processors among several crates reduces the contention problem if the data they need for operation is similarly distributed, but occasionally they will need access to one another's data.

In the Fastbus system, many segments operate independently whenever they can, and are temporarily linked together by segment interconnect modules when necessary for inter-segment data transfers.

The cables which interconnect backplane segments may be segments themselves, and may contain devices other than segment interconnect modules. Devices on cable segments obey the Fastbus protocols, but must provide their own power since power is not provided in the cable segments.

Segment interconnect modules monitor the activity on the two segments they connect, waiting for an address to appear which is in a set of addresses which they have been programmed to recognize. They respond to the address by requesting use of their other segment and asserting the given address on that segment when they gain control. The two segments remain locked together until the transaction is complete. An arbitrary number of segments can be linked as needed for a given transaction. The address contains all the information needed to direct the appropriate segment interconnects to respond and form the correct connection.

In order to use the address to provide the routing information in a practical way, the total address space available to the system is divided among the segments in such a way that the most significant bits of the address are sufficient to specify which segment is addressed. Every module on a given segment then has the same value for the high-order part of its address, and that value can be thought of as the segment number. The less significant bits serve to specify which module on the given segment is addressed, while the least significant bits specify the part or function within the module which is addressed.

The segment interconnect modules can thus be implemented in a simple way by using the high-order address bits to address an internal memory which contains a one in the locations corresponding to addresses which are to be recognized and passed, and zeroes in the other locations. When the system is initialized, these memories are loaded with the patterns needed to route all permitted transactions correctly.

With this scheme, there are no restrictions on the kinds of interconnections which may be made between segments. They may be connected in a tree structure with a big computer at the trunk and data acquisition modules at the leaves, for example. If a high traffic demand between two widely separated segments causes excessive interference with the intermediate segments, a cable segment can be added which bypasses the intermediate segments. No module address changes are required because of this change, and once the route tables in the segment interconnects are reinitialized to make use of the new route, the interfering traffic will disappear from the formerly intermediate segments. Tree, star and ring structures can all be accommodated by this scheme.

However, some rules must be obeyed when setting up the route table information for

the segment interconnects. For example, only one interconnect module may respond to any given address on a segment, since there must be only one path used for a given transaction. A procedure has been developed for creating the routing information automatically, which makes it easy to reconfigure the system as needs change. Dedicated systems which do not change may avoid the need for initialization by storing the necessary route information in permanent read-only memories.

THE FASTBUS PROTOCOL

The Fastbus uses a 32-bit parallel bus for address and data. It may seem strange to multiplex address and data in a system which strives for ultimate speed, but it turns out that the speed penalty is not nearly as great as is usually supposed, since the data cannot be used until address recognition is complete. The reduction in the number of bus lines, connections and transceivers results in significant economic and reliability advantages for multiplexing. In order to achieve maximum speed for data transfer, block transfer modes have been developed in which a single address word is followed by a number of data words. Speed has been further optimized by using every transition of the data strobe to transfer data in a block transfer. In block transfers, the time penalty for multiplexing address and data is insignificant.

To initiate a transfer, a master device asserts the slave's address on the 32 AD lines followed by the address strobe, AS. (A master is a device which initiates a transaction by acquiring control of the bus and asserting an address. A slave is a device which responds to the address on the bus. A module may be able to act as either a master or as a slave at different times.) The address word assertion sets up the path between master and slave, through segment interconnect modules if necessary. When the slave recognizes its own address, it responds with the address acknowledge signal AK. AS and AK remain asserted until the completion of the transfer, and serve to lock other users off the bus and cause them to ignore all bus activity. In fact, once the connection is established and the AS-AK lock is complete, the two modules could do almost anything with almost any other line on the bus and no other modules would be disturbed. In order to facilitate the construction of compatible modules, however, standard protocols for most useful transfers have been specified.

When the master sees the AK response, it knows that the address information is no longer needed, and removes it from the bus. It now asserts data and the data strobe DS, in case of a write operation, waiting for the acknowledging DK from the slave before removing the data. For a read operation, DS is asserted along with the read line RD. The slave responds with data and DK. The transfer ends when the master sees DK, removes its signals including AS, and the slave, seeing AS removed, removes its signals including AK.

The connection between master and slave is always handled as described above, with full handshaking at the beginning and end of a transfer. Even when block transfers are using both edges of the DS-DK pair, the transfers are fully protected by handshake.

However, it is possible to perform a block transfer without handshake between modules which can handle the same data rates. In the case of a write, the master simply asserts data words and DS transitions at whatever rate is appropriate, ignoring the DK responses. In effect, DS becomes a clock which the slave uses to find the data words in a synchronous transmission. Handshake protected transfers require each data word to be on the bus for at least two bus propagation delays, while the data flows to its destination and the acknowledge flows back to the source. When handshake response is not used, however, several data words could even be flowing through the bus transmission lines at once. Without handshake, data can be transmitted at the full bandwidth capacity of the medium.

In most cases, transfers will use full handshake protection. The handshake permits either party to pause for a moment if necessary (perhaps for refreshing dynamic memory chips), and allows either party to terminate a transaction early (in case a buffer overflows, for example) with both parties having full knowledge of how many words were successfully transmitted. Transfers without handshake require the master to know the capabilities of the slave and the bandwidth of the entire path in order to choose a workable DS clock rate. If the transfer does not push this rate near its limit, one could just as well have used handshake in the transfer and not had the worry.

The information as to whether handshake is to occur and whether a block transfer using both strobe edges is to occur is carried in two additional lines.

MIXED MODE TRANSFERS AND MACROS

The transfers described above can be generalized to allow data flow to reverse direction. For example, a read-modify-write cycle asserts address, reads data, turns the AD lines around again by removing the RD signal, and writes the modified data back to the slave. Such a transfer is uninterruptable by any other processor, since it is locked the whole time by the AS-AK lock and no other device can use the bus. It thus forms the kind of indivisible operation needed in multiple processor systems for coordinating use of shared resources.

The transfer can be extended to even more complex operations, so long as master and slave agree on the meaning of each bus cycle. For example, the address which connects master and slave could be followed by a data word which the slave interprets as an internal address or special command, followed by another data word which the slave interprets as data.

Transfers of this sort are referred to as mixed-mode transfers. Note that the individual data words usually will not be sent as a block transfer, since the possibility of turning the bus around between words is being maintained, and the time needed for that is provided by the alternate edges of the data strobes. However, block transfers may also be included within mixed mode transfers.

A still more general kind of transfer on the bus is called a macro. It consists of a series of transfers (possibly mixed mode), each involving an address cycle, between which the master does not release the bus for arbitration, so no other master can get control. This can be very useful for synchronizing a set of slave modules which are shared by several processors, so that a coherent set of operations can be performed without interference by other processors. This mechanism even works if the slave modules are all on different segments, as the segment interconnect is designed to maintain any connection until the master gives up its bus.

CONTROL AND STATUS REGISTER ADDRESSING

Certain registers and functions in modules need to be separated from the normal data registers in a way which provides some protection from accidental access and which does not interfere with the allocation of addresses to the normal data portions of the modules. For example, two memory modules should be able to have their addresses set so that the memories are adjacent in address space, allowing them to be used as one larger memory. However, they may contain control registers and status registers associated with memory protection or error detection and correction, and these registers must also be accessible in some way. Furthermore, it is desirable that modules have basic status and information registers in standard locations so that standard shared programs like the system initializer can find them easily.

The method chosen to accomplish this is a special case of the mixed mode transfer. The module is selected by its address, with additional information coded on other lines to specify that this is really a control/status access. The first data word is the number of the internal control/status register, and the second is the data to be read or written. This three-cycle transfer provides a full 32-bit address for use within a module, which is enough address space so that it can easily be allocated in standard ways without fear of a shortage. An abbreviated, two-cycle transfer is also being considered for the more common functions. It is hoped that at least the more complex modules will include a read-only memory containing information about the module and its properties which can be used by programs and people to make managing large systems easier. A large block of addresses within the module will be reserved for this purpose. Standard locations are also being specified for all the usual control and status bits.

BROADCAST TRANSACTIONS

A broadcast transaction is one in which a single master sends information to a multiplicity of slaves. Broadcasts can be used to synchronize modules or clear a bank of counters. Since more than one slave may be involved, no handshake between slave and master is possible. However, a system handshake has been devised which informs the master that his command has propagated to every segment to which it was addressed. The master asserts an address with other lines indicating that a broadcast is to occur. The address may refer to a specific segment or may refer to all connected segments in a pattern controlled by the route tables in the segment interconnects, or it may specify all segments beyond a given segment in that predefined pattern.

The general address used for broadcasting has zeroes in its most significant bit positions, so that the route table entries corresponding to zero address are used for routing broadcasts. For this entry, more than one segment interconnect may recognize the address and pass it onward, since no handshakes are to be returned. The pattern formed by the pathways propagating from the broadcast master must form a simple tree with no cross connections, another rule to be applied by the initialization program.

When the broadcast address has propagated successfully throughout the system, the system handshake occurs and the master asserts the control register number it wishes to broadcast to, following the protocol of the control/status register addressing discussed above. When the system handshake is returned, the control data is asserted. Thus, any kind of standard control operation may be performed at once on a large set of modules by a broadcast.

Broadcasts may take some time to start, since they must wait for all conflicting use of the segments involved to complete. Applications requiring very fast response may have to resort to separate cables, since there is no way to achieve fast response in a reliable way in a multiple segment system. Once the system connection is complete, however, speed of execution of the data cycles is limited only by system signal propagation delays, so a reasonably synchronous execution of the command is achieved. At least one can be certain that all modules will see the command before any other bus operations will occur.

SPARSE DATA SCANS AND THE "T" BUS

The Fastbus design includes an auxiliary 32-line bus on the backplane, called the "T" bus, which is included in the main connector. Also in the main connector are a 33d line called ET (Enable T) and a 34th pin, called T, which is not bussed.

Inside the backplane, the T pin is connected to the T bus line corresponding to the module position number. Thus, a module in position 12 finds its T pin con-

nected to T bus line 12, etc. The T bus thus can be used for positional information, and combines the functions of the CAMAC N and L radial lines. Since the bussed lines appear in all module positions, there is no preferred or dedicated controller position in a crate. The ET line controls the direction of information flow through the T pin.

The T bus was originally included in the Fastbus to provide a means for rapidly scanning sparse data in detector front-end modules. A controller on the backplane broadcasts a command to the front-end modules which causes them to assert their T pins if they contain data. The resulting pattern on the T bus shows the controller immediately which modules need to be read out, thus avoiding the overhead of polling them one at a time. The T bus can then be reversed by means of the ET line and used by the controller to cause the modules to put their data on the AD lines one after another without going through a normal address cycle.

Most modules ignore the T bus except for the T pin, which is used in geographical addressing and in a simple interrupt mechanism. Use of the T bus is always under control of the bus master, and in general a master and slave on the same backplane may use the T bus during a transaction in any way they wish without disturbing any other modules. The T bus exists only on backplane segments. Its geographic addressing function must be simulated in modules located on cable segments.

GEOGRAPHICAL ADDRESSING

In systems of any significant size, the effort involved in setting all module addresses by manual switches is too great. Furthermore, some means of automatically determining the locations of modules is needed, if only for use as a record or a check. When a system without switches is first turned on, the control registers which will contain the module address information are initialized randomly. Some other mechanism is needed to address the modules in order to load the module address register with the proper contents.

An addressing mechanism has been included in the Fastbus which allows accessing a module by its location in the system, or by its "geographical address" rather than by its normal or "logical address". The geographical address of a location in a segment is the position number of the location. The address is recognized by circuitry on the arbitration master on the bus, and passed to the appropriate module by means of the T pin of the T bus.

The geographical addressing mechanism can be used to access the control and information registers in uninitialized modules so that an automatic procedure can be used to initialize them.

INTERRUPTS ON THE FASTBUS

An interrupt is a request from some device to some processor for service or attention. Since interrupts may have to cross segment boundaries, and since they must carry information, they are handled by normal Fastbus protocol transactions. The interrupting device addresses an interrupt sensing control register in some processor, and writes his own address and possibly other information into the register. The processor then has all the information it needs to find the interrupting device and service it at some later time.

In some systems, large numbers of simple modules may need to signal a request for service without having the capability of gaining bus mastership and performing an interrupt write. Within a single segment, such modules may assert a service request (SR) line, which can be monitored by a special interrupt service module which does have the circuitry to gain mastership and find the requester, whether by means of the T bus or by polling or some other means. The interrupt service

module may then perform the necessary service itself, or it may send a normal interrupt message on behalf of the simple requester to some other processor.

ARBITRATION FOR FASTBUS MASTERSHIP

Since multiple devices on a segment may wish to become master of the segment in order to perform some transaction, some means is needed to prevent more than one of them from using the bus at one time. Ten lines on the bus are dedicated to the solution of this problem. Six of the lines are used to hold a priority code which determines which competing device wins mastership, while the other four are used to synchronize the requests. The arbitration mechanism operates in parallel with use of the bus, so that little time need be wasted in switching from one master to another.

At a given time, each requester tries to assert his priority on the AL (arbitration level) lines. The lines perform a "wire-or" function, so that any asserting requester overrides nonasserting ones at each bit position. Each requester compares his code with the code on the AL lines bit by bit, from most to least significant. If he sees an AL line asserted which he did not assert, he removes his assertions of all less significant bits, because he knows that a higher priority requester is competing. After four bus propagation delays, only the highest priority code remains asserted and each requester knows whether he has won or lost.

Of the 64 possible codes, zero is not used because it is easily confused with an idle bus, 1 through 31 are available for use within the segment, and 32 through 63 are used as "super" priorities which must be assigned uniquely throughout an entire connected system. The normal priorities 1-31 must be assigned uniquely to modules within a given segment, but exist on every segment to be used over and over again. When a segment interconnect connects a master to another segment, the priority level used for arbitration on the second segment will normally be that of the segment interconnect module rather than that of the originating master. However, if one of the super priorities was used by the master, the segment interconnect will propagate that priority onto the second segment, which it is free to do since the super priorities are unique within the system. The super priorities can be useful in preventing undue delay for important broadcasts, and can help expedite important messages, which otherwise may suffer from fluctuating priorities as they form paths through the system.

The current master determines when he will be finished with the bus, and releases the arbitration circuitry so that the next master can be selected before he finishes. He thus maintains ownership of the bus as long as he likes, which is the mechanism used to implement the macro transactions discussed above.

CONCLUSION

The Fastbus design has evolved over a two-year period into a simple, clean and inexpensive system which can solve a broad spectrum of problems. Its ability to cope with extremely fast as well as slow devices, its easy expandability, its parallelism, modularity, and multiprocessor support commend it for a wide range of applications.

The status of the Fastbus is reported annually at the IEEE Nuclear Science Symposium. For example, see the article by R. S. Larsen in the IEEE Transactions on Nuclear Science, Vol. NS-26, pages 679 through 685, February 1979.

For current information, contact Louis Costrell, Chairman, U. S. NIM Committee, National Bureau of Standards, Center for Radiation Research, Washington, D. C. 20234, telephone (301) 921-2518.