DOE/ER/80454--19

THE BULK-STORE MODULE FOR FASTBUS MEMORY APPLICATIONS

The Results of a Phase II SBIR Grant

Scientific Systems International, Ltd.*

Work performed under the auspices
of the U.S. Department of Energy
SBIR Contract DE-AC02-87ER-80454

*3491 B Trinity Drive
Los Alamos, New Mexico
(505) 662-6712

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

TABLE OF CONTENTS

1. Overview	1
2. Chronology of the Contract	2
2.1 Actions by ASIC Manufacturers	2
2.2 Organization of the Report	3
Appendix A - Bulk Storage Module Specifications	4
Appendix B - Bulk Storage Module Schematics	34
Appendix C - FASTBUS Slave Interface Specifications	92
Appendix D - FSI Schematics, AMCC BiCMOS Array	110
Appendix E - FSI Schematics, Raytheon ECL Array	173
Appendix F - FSI Technical Paper and Brochure	208

THE BULK-STORE MODULE FOR FASTBUS MEMORY APPLICATIONS FINAL TECHNICAL REPORT

1.0 Overview

The FASTBUS system specification for high-energy physics and other data system applications defines a large address space in which support modules may reside on connected segments. Experiments in the physical sciences using FASTBUS indicate that the spectrum of support modules must continue to grow and be extended in capability. The features of the FASTBUS protocol can be effectively used to help increase the available data storage space and improve performance for large-memory configurations such as the Bulk-Store Module (BSM). To this end, the module design features automatic address-only histogramming, memory interleaving at the module level, and circular data buffering as well as normal random memory access.

This report describes the components of the 32-bit, 8 megaword error-correcting memory module for the FASTBUS data system designed under a Phase II SBIR grant through the U. S. Department of Energy, Division of Energy Research. A key feature of the BSM is the design of a single-chip application-specific integrated circuit (ASIC) to interface the FASTBUS protocol to any slave application, and particularly to the BSM. The FASTBUS Slave Interface (FSI) design contains features that help the BSM perform its specified tasks, in addition to providing a complete interface to the FASTBUS protocol, including broadcast and advanced mode response.

We further cover the chronology of the Phase II work effort and our experiences with two ASIC manufacturers in attempting to complete the actual fabrication of the FSI chips.

The appendices of this report contains the functional description and design schematics of the BSM, and of the FSI ASICS in two different semiconductor technologies.

2.0 Chronology of the Contract

The SBIR Phase II portion of the Department of Energy contract DE-AC02-87ER80454 (proposal no. 5179-87-II, The Bulk Storage Module for FASTBUS Memory Applications—BSM) was approved to begin on December 1, 1987. The 120 day advance funding period was concluded on March 31, 1988. Design work on the BSM memory section, control logic and FASTBUS Slave Interface (FSI) extended through October 1989. ASIC application testing for test vector generation was completed in April 1990. Simulation testing of the integrated design was completed in November 1990. Delays from Raytheon Corp. in completing the workstation design kit delayed the fabrication of the FSI interface chip and necessitated a request for a 6 month extension through March 31, 1991 and again through September 30, 1991, and an additional extension through March 31, 1992 was granted. Further difficulties with pre-layout simulation at Raytheon finally caused Raytheon to admit that they could not complete our design and would terminate the sub-contractual agreement in early December 1991. We restructured the original contract with DOE, and a one year extension to March 1993 was granted. The modified contract eliminated the FASTBUS memory module while retaining the slave interface chip. The contract for fabrication of the FSI ASIC was shifted to Applied Micro Circuits Corp. in San Diego, California. The FSI design package and test vectors were modified to AMCC circuit elements, and this modified design package was been sent to AMCC in late August 1992. In late October, AMCC conveyed to us that they would not accept the design and would not, without a new contract with additional NRE fees, accept further designs in the BiCMOS array technology. Following submission of the FSI design package in late August, we were told by AMCC staff that, owing to difficulty with yield in the particular array technology, they would not accept our design. In actual fact, AMCC unilaterally canceled our contract and refunded the NRE paid (\$28,000.00). Following these discussions with AMCC, and discussions DOE-Washington, we concluded that it was no longer feasible to attempt completion of the fabrication of the FSI prototype circuits, but to conclude the Phase II contract with the designs of the BSM and the FSI in two technologies. The final progress report stated this conclusion on March 31, 1993.

2.1 Actions by ASIC Manufacturers

The actions by Raytheon Company in December 1991, and by Applied MicroCircuits Corporation (AMCC) in December 1992, have made it impossible for us to complete the FSI ASIC and thus the final work item of the Phase II contract (6.2.8). While the semiconductor companies were

initially very anxious to accept our designs, we found that internal problems with the technologies and the business outlook for ASICs in general made small business, and especially small business serving the scientific community, expendable. Our recommendation, based upon these devastating experiences, is that the physics community and small technology- businesses serving this community should be very wary of attempting to use any of the faster, more complicated integrated-circuit technologies.

Within the physics community, there have been some successes with CMOS custom and semicustom designs, and one example of a successful small ECL ASIC, but large (>20,000 gate) ECL or BiCMOS designs are rare to non-existent. Unless the potential market is very large (e.g., 100,000 units per year fabricated), even the government does not have a good chance of being able to complete a design. We further found, in the case of Raytheon Company, that the tools available to the company were out of date and quite antiquated. Their eventual exit from the ASIC business could be a reason why their design tools were poor. In the case of AMCC, the design tools were adequate, but the company seemed to be unable to meet specifications with the BiCMOS technology and were actually moving to a bipolar ECL technology when they canceled our design agreement. In both cases, the small size of the potential FASTBUS market was a deciding factor to drop our agreements.

2.2 Organization of the Report

The designs for the entire contract are complete and contained in Appendix A (BSM specifications), Appendix B (BSM design), Appendix C (FSI specifications), Appendix D (FSI design-Raytheon), and Appendix E (FSI design-AMCC). Appendix F contains a copy of an IEEE Nuclear Science Symposium paper on the FSI and an advertising flyer for the FSI that describes the marketing information that would have gone to the experimental physics community had we been able to complete its prototype fabrication.

APPENDIX A
Bulk Storage Module Specifications

1.0 Introduction

This document presents application information for a FASTBUS slave module, the Bulk-Store Module (BSM). As described within the following sections, the BSM is a flexible, highly configurable system component. It is capable of performance at rates twice the normal FASTBUS transaction rates by interleaving at the module level, and offers storage capacities equivalent to large magnetic disks. In addition to its role as a random access memory, the BSM may serve as an auto-incrementing histogram memory or as a circular buffer for event list-mode storage. All this functionality is available through standard FASTBUS transactions.

The combination of four principal characteristics distinguish the BSM from other memory implementations. First, BSM modules can be configured as either a set of autonomous modules or as a single "logical" BSM consisting of the set of cooperating BSM subsystems (or subdivisions, the term for a BSM module operating in this mode). In the latter mode the modules are said to be "interleaved". Second, the BSM offers dual-ported memory, one port on the crate segment and the other accessed via standardized protocol at the auxiliary connector. Third, the memory array in the BSM may function as an auto-incrementing histogram memory. In this mode the BSM is addressed with a Null Operation which transfers no data. Fourth, the memory array may function as a circular buffer for event list-mode storage. This mode offers buffering for events consisting of an arbitrary number of FASTBUS words at full FASTBUS speeds.

As a conventional memory unit, the BSM module supports large memory capacity and reliable operation through a high level of circuit integration. The BSM utilizes the latest high density DRAM chips to achieve a memory capacity greater than or equal to 8 million FASTBUS words, i.e. 32 MBytes. The actual memory array includes additional memory for error detection and correction thereby ensuring memory data integrity. The BSM seeks to provide the largest memory possible (8 MWords or more with error detection and correction) with existing memory technology and with performance compatible with FASTBUS operations. Recognizing the significantly higher memory density available with dynamic RAM chips, the BSM requirements assume an implementation with these components.

1.1 Module Interleaving

Traditionally, such implementations have employed interleaving banks of segmented memories to achieve effective memory speeds comparable with the faster (but less dense) static RAM chips. The BSM substitutes module interleaving for the traditional bank interleaving. Standard FASTBUS protocols are employed to achieve the module interleaving in a **totally transparent manner**. Special care is taken to ensure synchronization between the cooperating subdivisions which comprise the "logical" BSM unit. While the subdivisions take turns in responding to FASTBUS transactions, a single subdivision drives the AK and DK protocol lines to provide "glitch-free" operation.

Communication between the BSM subdivisions is accomplished *via* one interconnecting front-panel signal. Without the complexity of bank interleaving internal to each memory module, board space requirements are diminished so that other significant new functions can be included in the design.

While the BSM module is generally expected to operate in the interleaved mode with multiple modules to achieve higher performance, this is not required. For single module applications, page access to the dynamic RAMs is utilized to achieve high performance for block mode transactions independent of interleaved mode. Control is accomplished through bits written to CSR#0 and CSR#1. In addition the interleaved mode is fully independent of the memory access mode which allows selection of random access memory, histogramming memory, or circular buffer access.

Module interleaving is limited to clusters of 2, 4, 8, or 16 modules. Of course, other memory sizes may be configured by grouping interleaved clusters together, *i.e.* grouping an interleaved pair of modules with an interleaved set of four for a total of six BSM modules.

1.2 Dual-ported Interface

Generally, more than one memory user exists in a typical memory system environment. For memory users on the FASTBUS crate segment, this shared access is achieved through protocol provided to bus users by the FASTBUS arbitration logic. Often this sequential access is quite restrictive for the dynamic requirements of a data acquisition system.

To improve performance, FASTBUS memory systems have been built with two memory ports: one for a crate segment and another for a cable segment. This configuration allows data from the crate segment to be buffered within the memory module and subsequently read by a centralized master with access from the cable segment. Data flows in a single direction from the crate segment into the memory and upward into the system hierarchy through the cable segment. To accommodate such configurations requiring dual-ported memory, the BSM supports an alternate FASTBUS port interfaced to the auxiliary connector. Here, the interface signals are those for the standardized auxiliary connection to an external rear-mounted cable segment driver.

The use of this standardized auxiliary connector for cable segment implementations (see Section 16.2 of the December 29, 1987 Addendum & Errata to FASTBUS Standard ANSI/IEEE Std 960-1986) provides additional flexibility for the system designer. With a minimum of difficulty, personality boards can be designed to allow connection to an arbitrary memory bus. Thus the system designer can, as technology delivers ever more powerful processors, introduce high performance processors at the front-end of data acquisition systems with a connection to the BSM to accomplish data compaction, rejection, and reduction.

1.3 Histogram Capability

The BSM offers a histogramming capability at full FASTBUS speeds. An address-only Null Operation is used in the histogram mode to increase the effective throughput. With the histogramming mode enabled, the BSM treats each such null operation as a virtual increment command. The address is pushed into a FIFO and incremented as time permits. This feature, coupled with increasingly powerful processors embedded within the FASTBUS environment, may eliminate the necessity of recording all events during an experiment for later off-line reduction. The large data space afforded by the BSM memory enables significant multiparameter storage of sufficient resolution to allow critical cuts within the data space to be made off-line.

1.4 Circular Buffer Capability

The BSM offers an event list-mode capability at full FASTBUS speeds when used in conjunction with block mode transactions. Upon receipt of a block of data while the circular buffer mode is enabled, the BSM skips one or more locations in the memory for later storage of control information. The event parameters are then stored in the circular memory. After the last word in the block is received, the event length is inserted as part of the control information. Block read operations move only the parameters associated with a single event. Use of this feature in a BSM acting as a buffer for an asynchronous processor allows rapid clearing of the front-end electronics followed by the less time-critical data manipulation process. In general such a calculation results in either scheduling one or more histogram channels for incrementing or passing a derived parameter set forward for subsequent storage (and processing) within another BSM or a combination of both.

In the circular buffer mode, data flows into memory from the designated source port, into memory, and out the destination port. With two interface ports, the BSM naturally supports only two directions for this data flow. The direction is defined in CSR#0.

When operating in the circular buffer mode, memory in the BSM module is accessible in two ways. Access through CSR space (CSR#10) presents a FIFO-like interface with CSR#11 and CSR#12 acting as pointers to the event information. Random data cycles in CSR space may manipulate the pointers subject to some restrictions. Block transfers to CSR 10 moves data to/from the memory with the CSR 11 or 12 acting only as pointers, identical in function to the conventional NTA register. Access through data space is still possible and can be meaningfully used by an intelligent processor that does not care to process data sequentially.

1.5 Configurability

By limiting the operational features of the BSM, the module may effectively serve a broad spectrum of experiments in a cost effective manner. Special requirements may demand the introduction of highly system dependent capabilities optimized for specific applications. For example, we envision that the BSM may be easily integrated as the bulk memory element of an "event oriented" system by addition of a custom front-end module

designed to save blocks of information labeled by event number. Such a custom controller might buffer the block received from the user, select its storage location within the BSM, record this address in some directory structure, and move the block into BSM data space via the auxiliary port. Subsequent access of the block from the crate segment side requires a user to first recall the address for the event in the BSM through access to the directory in the controller, followed by BSM access.

1.6 Compliance with FASTBUS Specification

The BSM conforms to the mandatory specifications of ANSI/IEEE Standard 960-1986 for both interleaved and non-interleaved applications. For the latter case, the module behaves as required for traditional slaves. For the the interleaved application, the logical BSM designates one of the subdivisions as the driver for the AK and DK slave responses. By providing a single source for these protocol lines, it is possible to ensure "glitch-free" operation in a voltage-driven bus environment.

2. Module Implementation

2.1. Form Factor

The BSM is a single width FASTBUS module.

2.2. Front Panel Indicators

Front panel indicators are limited to two yellow LEDs each of which is lit when the module is acting as an attached slave to its corresponding interface port (see Section 13.5 of the ANSI/IEEE 960 Std 960-1986).

2.3. Front Panel Connectors

Front panel connectors are limited to two SMB connectors. An output connector for the output response control signal (ORC) allows the wire-oring of the ORC signals of all participating subdivisions of a logical BSM module. This wire-ored signal is used as input to the input connector for the input response control signal (IRC) in one and only one BSM module.

2.4. Internal Jumpers

A jumper option on the BSM selects the auxiliary bus mode. If set = 0, the bus is bidirectional, and column A is bidirectional and column B should be ignored. If set = 1, the bus is unidirectional (as for cable segment connections), and column A is used for BSM unidirectional inputs and column B for unidirectional outputs.

A jumper option on the BSM selects the busy-response mode. If set = 0, then WT = 1 is asserted if the other interface port is active, or if data space access is attempted while the stack of addresses stored in the increment FIFO is not empty and the histogramming mode is enabled. If set = 1, then SS = 1 is asserted if the other interface port is active, or if data space access is attempted while the stack of addresses stored in the increment FIFO is not empty and the histogramming mode is enabled.

A jumper option on the BSM defines the size of the data space memory array. The options includes memory sizes of 8, 16, 32, 64, and 128 Mwords.

3. Sample Applications

By way of example, three applications are presented in the following sections to illustrate the use of the BSM module. First, a single module configuration is presented followed by a pair of examples with four module configurations. In each case the interconnections for hardware are discussed. Section 5 lists FASTBUS "programs" used to initialize the hardware.

3.1 Single module hardware configuration.

In our first example, a single BSM is used in a simple experiment. Figure 1 illustrates the configuration that utilizes a serial to FASTBUS protocol converter card behind the FASTBUS backplane at the auxiliary connector. It must be emphasized that at this time no such serial to FASTBUS protocol converter exists, but an adequate model could be the buffered interconnect discussed in the literature and referenced within the FASTBUS specification itself. A single connection is made on the front of the BSM to tie the ORC and IRC connectors together. No other connections are required.

Fig. 1. Configuration for BSM used with serial protocol converter.

The data path employed in this example expects data to flow into the BSM from a detector subsystem. Upon receipt of a block of data from the detector, the protocol converter addresses the BSM without arbitration (since it is the sole user of the port), and initiates a block transfer. When all data has moved in the circular buffer, the connection is broken to signal the end of the record. Data captured within the circular buffer mode is later read out through the crate segment for processing.

3.2 Four BSMs with processor module configuration.

In our second example, a set of four BSMs is used in conjunction with an independent processor module. Figure 2 illustrates the configuration. First, note that the four BSMs have their ORC outputs bussed together, and then tied to the IRC input connector of the right-hand unit. In this manner, the right-hand unit is designated the protocol-response generator, and it drives the AK and DK responses for the BSM foursome. Also note the private memory bus connecting the BSM set with the processor module. Like the protocol converter of the previous example, the processor module is the only master on this private bus, and thus may omit any arbitration cycles for access to the memory via this bus.

Fig. 2. Configuration with four BSMs used with processor module.

Although the processor may exploit all the access modes afforded by the BSM, the intent of this example is to illustrate only the extended RAM application. Hence the data flow is rather simple; single words flow to and from the BSM, or, if desired, blocks move to and fro. Slot address are hardwired at the auxiliary connectors, and define the geographic addresses to be 0, 1, 2, and 3. The FASTBUS crate segment port is available to other masters to insert or remove data from the BSM data base.

3.3 Configuration with four BSMs between crate and cable segments

In our third example, a set of four BSMs is used as a large buffer with access from the cable and crate ports. Figure 3 illustrates the configuration. As in the previous example, the four BSMs have their ORC outputs bussed together, and then tied to the IRC input connector of the right-hand unit. Again, the right-hand unit is designated the protocol-response generator, and it drives the AK and DK responses for the BSM foursome. While the signal source for voltage-driven signals is essential on the crate port, it is an unnecessary but supported feature for the current-driven implementation of the cable. The auxiliary cable interfaces provide unique geographic address on the cable for each BSM module. Unlike the the previous examples, arbitration cycles for access to the memory are required on each bus.

Fig. 3. Configuration with four BSMs used with a conventional cable segment.

Memory users on either bus have complete access to the various modes supported by the BSM. The modules could be used as a large circular buffer memory as in example 1, or as a large shared memory as in example 2. The configuration for the application is defined by the user via software control. No software "program" is presented in Section 5, however, since initialization is little more than a combination of examples 1 and 2.

Note that the logical address on the cable port need not be the same as on the crate port. Systems consisting of many crate systems tied together with a common cable segment can exploit this feature by retaining identical logical address assignments within the crate or crate cluster, but providing unique logical address at each cable port associated with the participating BSM systems.

4. Operational Details

The BSM is a FASTBUS slave module conforming to the mandatory specifications of ANSI/IEEE Standard 960-1986.

4.1 Interface

For the the crate segment, signals implementation all FASTBUS operations from chapters 4, 5 and 10 of ANSI/IEEE Standard 960-1986 except the use of SR and TP. If parity is enabled by the master (PE), then parity is checked for address and write operations while parity is generated for read operations.

For the Auxiliary Connector (Cable Segment), signals comply with Table 16.2(b) of the 18 May, 1987 Addendum & Errata to FASTBUS Standard ANSI/IEEE Std 960-1986, Recommended Utilization of Auxiliary Connector for Cable Segment Implementations. Like the crate segment signals, these signals implement all FASTBUS operations from chapters 4, 5 and 10 of ANSI/IEEE Standard 960-1986 supported except the use of SR and TP.

A jumper option on the BSM selects the auxiliary bus mode. If set = 0, the bus and column A are bidirectional and column B is ignored. If set = 1, the bus is unidirectional (as for cable segment connections), and column A is used for BSM unidirectional inputs and column B for unidirectional outputs.

4.1.1 SS Response Codes at address time with AK(u)

If MS = 0 selecting data space or MS = 1 selecting CSR space, and if the BSM recognizes the device address or geographical address, then the BSM generates SS = 0.

No logical address can generate SS = 7. If the address is for CSR space, the internal address (IA) within the module is meaningless and results in no action. If the address is for data space, the IA is fully decoded and hence cannot produce an error.

4.1.2 WT at address time

The Wait (WT) signal is set WT = 1 if all conditions are satisfied for the BSM to attach to the master but the stack of addresses in the increment FIFO is full, the address cycle is directed to data space, and the histogramming mode is enabled. When the stack is no longer full, the BSM sets WT = 0.

4.1.3 SS response codes at data time with DK(t)

Unless a data error is detected, SS = 0 is issued.

SS = 1 is asserted in two circumstances if the busy-response-mode option is jumper selected to enable the SS = 1 response mode. The master should expect successful access very soon, and should continue its attempts for access. First, SS = 1 is asserted if the other interface port is active. Second, SS = 1 is also asserted if data space access is attempted while the stack of addresses stored in the increment FIFO is not empty and the histogramming mode is enabled.

If end of block is asserted, the master must disconnect and try again or execute a secondary address cycle. The end of block is signaled with SS = 2 during a block transfer when an attempt is made to transfer another data word following the transfer of data from the last valid location. Also when the circular buffer mode is enabled, circular buffer full and circular buffer empty generate SS = 2 for write and read block transfers, respectively. The master may continue with another block transfer if desired.

The SS = 6 response indicates that the slave is unable or unwilling to accept or assert data because of an error associated with 1) the data, 2) the current value of the IA not valid for CSR space (see Section 3.3.1), or 3) an MS value where $MS > 3$. No changes have taken place in the BSM except for issuance of the response. The SS = 6 response is not issued for secondary address cycles. See Section 5.3.2 of IEEE 960 for master response.

The SS = 7 response is issued for the same reasons as SS = 6. The difference is that the BSM asserting SS = 7 has accepted the faulty data or is asserting data known to be in error. SS = 7 is asserted if the next transfer address (NTA) is loaded with an invalid internal address (IA) during a CSR secondary address cycle. For the BSM this may happen only in CSR space since the internal address data space is fully decoded. See Section 5.3.2 of IEEE 960 for master response.

For data space read cycles SS = 7 is given when the data cannot be corrected. For data space write cycles that detect a parity error, SS = 7 is given and the data saved with syndrome bits derived from the bad data.

4.1.4 WT at data time

WT = 1 is asserted in two circumstances if the busy-response-mode option is jumper selected to enable the WT = 1 response mode. The master should expect successful access very soon, and should continue its attempts for access. First, WT = 1 is asserted if the other interface port is active. Second, WT = 1 is also asserted if data space access is attempted while the stack of addresses stored in the increment FIFO is not empty and the histogramming mode is enabled. When the stack has been emptied, the BSM sets WT = 0.

4.2 Functional Features

All FASTBUS operations from chapters 4, 5 and 10 of ANSI/IEEE Standard 960-1986 are supported except for use of SR and TP. If parity is enabled by the master (PE), then parity is checked for address and write operations, and parity is generated for read operations.

4.2.1 Module Interleaving

BSM modules operate in either an interleaved or non-interleaved mode. In the non-interleaved mode the BSM module is an independent slave module. In the interleaved mode, two or more BSM modules act as a single "logical" unit while individually performing as independent subdivisions.

In addition to control of logical addressing, CSR#0<01> controls the interleaving mode of the BSM. If this bit is set to 0, then the BSM operates in the non-interleaved mode. If set to 1 and if CSR#1 differs from zero, then the BSM operates in the interleaved mode. No other control is required.

The module interleaving mode will not change once the BSM module subdivisions are selected as the attached slaves. The AS-AK lock must be dropped and reconnected to effect a change of interleaving mode.

Geographical addressing is used to initialize the BSM modules. For each BSM module, CSR#3 is loaded with the logical address. If the interleaved mode is used, then the logical address includes the least significant bits that must be unique for each subdivision within the "logical" BSM. CSR#1<03:00> is loaded with the bit mask identifying which of the least significant bits of CSR#3 are interpreted for subdivision selection if the interleaved mode is used, i.e. CSR#1<03:00> is not 0. The use of these fields in CSR#1 and CSR#3 is illustrated in fig. 1. By setting CSR#0<01> = 1 after loading CSR#3 and CSR#1, the BSM module is enabled for logical addressing and enabled for interleaving if CSR#1<03:00> is not 0.

Fig. 1. Fields required to specify logical addressing with subdivision address, and depth of interleave.

The scope of the connection between a master and a BSM module depends on the interleaving mode of the BSM and addressing mode of the master. If the BSM is geographically addressed or if the interleaving mode is disabled, the master connects to only the selected BSM module. If logically addressed and if interleaving is enabled, then the master first connects with the selected BSM module but connections with other subdivisions are possible *via* subsequent secondary address cycles or block transfer cycles.

Once a master has attached to a BSM, the BSM can not alter its mode of interleaving. For example, if the BSM is geographically addressed with logical addressing disabled, and the master enables logical addressing, the BSM unconditionally remains in the non-interleaved mode.

For certain modes and operations, transparent interleaved connections to CSR space are supported. During the CSR address cycle, the least significant bits of the logical address as defined in CSR#1 is used to establish a connection with the appropriate BSM subdivision containing the corresponding least significant bits in CSR#3. Thereafter, the connection is fixed to that BSM subdivision unless block transfers are performed with the circular buffer mode enabled.

Following a CSR address cycle and connection, all CSR space is accessed with data cycles as required in IEEE 960 except that all BSM subdivisions with identical CSR#3 contents exclusive of the bits selected in CSR#1<03:00> respond to write operations as if connected. For read operations, the selected BSM subdivision provides the data from the specified source. Exceptions apply to CSR#3 and to block transfers. Once the selected BSM subdivision has either accepted or asserted data, this subdivision triggers DK toggle.

Block read and write operations in CSR space are valid for only CSR#10. During these operations CSR#10 operates like a FIFO accepting and asserting data without advancing the NTA register. As discussed below, CSR#11 and CSR#12 increment at DK(t) for each data transfer cycle for read and write cycles, respectively.

During block read cycles of CSR#10 at DS(t) the bits of CSR#11<03:00> as defined in CSR#1<03:00> are used to establish a connection with the appropriate BSM subdivision containing the corresponding bits in CSR#3<03:00>. At DS(t) the selected BSM subdivision asserts the data bus contents. Then at DK(t) all BSM subdivisions with identical CSR#3 contents exclusive of the bits selected in CSR#1<03:00> increment by one the contents of CSR#11 as if connected. Any subsequent data cycle selects the BSM subdivision specified by the revised CSR#11, i.e. the bits of CSR#11<03:00> as defined in CSR#1<03:00> are used to establish a connection with the appropriate BSM subdivision containing the corresponding least significant bits in CSR#3.

During block write cycles of CSR#10 at DS(t) the bits of CSR#12<03:00> as defined in CSR#1<03:00> are used to establish a connection with the appropriate BSM subdivision containing the corresponding bits in CSR#3<03:00>. At DS(t) the selected BSM subdivision accepts the data bus contents. Then at DK(t) all BSM subdivisions with identical CSR#3 contents exclusive of the bits selected in CSR#1<03:00> increment the content of CSR#12 as if connected. Any subsequent data cycle selects the BSM subdivision specified by the revised CSR#12, i.e. the bits of CSR#12<03:00> as defined in CSR#1<03:00> are used to establish a connection with the appropriate BSM subdivision containing the corresponding least significant bits in CSR#3.

For most BSM modes and operations, transparent interleaved connections to data space are also supported. During the data space address cycle, the bits of the logical address as defined in CSR#1<03:00> are used to establish a connection with the appropriate BSM subdivision containing the corresponding bits in CSR#3<03:00>. Like CSR transactions, it is possible to alter the connection *via* secondary address cycles or block transfers.

Following a data space address cycle and connection, all data space is accessed with data cycles as required in IEEE 960 except that all BSM subdivisions with identical CSR#3 contents exclusive of the bits selected in CSR#1<03:00> maintain identical contents of the NTA register as if connected. Thus NTA changes during secondary address write operations and all block transfers cycles are propagated concurrently to all participating subdivisions. At DS(t) the bits of the NTA<03:00> as defined in CSR#1 are used to establish a connection with the appropriate BSM subdivision containing the corresponding least significant bits in CSR#3, and the selected BSM subdivision asserts or accepts the data bus contents. Then this subdivision triggers DK toggle.

For block cycles at DS(t) the bits of the NTA as defined in CSR#1<03:00> are used to establish a connection with the appropriate BSM subdivision containing the corresponding bits in CSR#3<03:00>. The selected BSM subdivision asserts or accepts the data bus contents. Then at DK(t) all BSM subdivisions with identical CSR#3 contents exclusive of the bits selected in CSR#1<03:00> increment the content of the NTA register as if connected. Any subsequent data cycle selects the BSM subdivision specified by the revised NTA register.

For any address or data cycle, the selected subdivision drives the SS lines for an address connection at AK and for a data cycle at DK. If the data cycle is a read cycle, then the selected subdivision asserts data on the AD lines. Unlike the AK and DK protocol lines, these lines are not required to be free of glitches.

To avoid possible glitches on the voltage driven crate segment, only one subdivision, the protocol-response unit, asserts AK and DK. When a subdivision within the logical BSM wishes to drive AK or DK, it generates a command to the protocol-response unit on the output toggle control line (ORC). The ORC signal from each subdivision is wire-ored together with all others of the "logical" BSM. This collective signal is connected as an input to the input toggle control (IRC) of a single subdivision, the protocol-response unit, that generates AK and DK.

Communication with the protocol-response unit is accomplished with a sequence of pulses generated by the various participating BSMs. If AS = 1, then the first command on the ORC line indicates that AK must be generated. Thereafter a command on the ORC is interpreted as a DK toggle command. When AS is set to 0 at disconnect time, then the protocol-response unit generates AK = DK = 0.

A command on the ORC line is initiated by setting ORC = 1. When the requesting subdivision detects the appropriate AK or DK response, it returns ORC = 0. The ORC signal is ECL 10K or 10KH compatible. The ORC line connects to a front-panel SMB connector. Participating subdivisions must be cabled together.

The collective ORC signal is delivered to the IRC connector of the BSM selected as the protocol-response unit. The IRC line connects to a front-panel SMB connector. Although each subdivision has an IRC connector and internal logic for AK and DK generation, only one may be connected at any one time. The IRC signal is ECL 10K or 10KH compatible, and is terminated in 100Ω resistor.

4.2.2 Dual-ported Interface

The BSM supports two FASTBUS ports. Each port interfaces FASTBUS signals and protocols on one side to memory access signals and protocols on the other. The crate segment drives one port and the auxiliary connector (cable segment) the other. Service is provided on a first-come, first-serve basis. Port arbitration is invoked if requests are simultaneous.

The port priority is defined by CSR#0<11>. If set = 0, then the crate segment has priority over the auxiliary port. If set = 1, then the auxiliary port has priority over the crate port.

For each memory access, a port presents a memory address, read/write option, and then requests memory access. When data is available for the requesting port, the port is signaled with output data available. If the transfer is a write operation, then the port must supply a write strobe to complete the cycle. A memory busy signal indicates that memory cannot yet accept another memory request. Arbitration is on a word basis only, and is transparent to each port.

Port lockout is governed by CSR#70<00>. If set, the port has exclusive access to the memory array. If an attempt to set bit 00 = 1 is performed while another port has exclusive access, a response SS = 6 is generated. Each port has a separate port control register, CSR#70

4.2.3 Histogramming Mode

In the histogramming memory mode, the BSM accepts special null operations as increment commands. The NTA latched during the address cycle is the address of the memory location to increment. The histogram mode is enabled by setting CSR#0<08:06> = 1.

Initialization for the histogramming mode requires only a few simple steps. Geographic addressing is used to load CSR#3 with the logical address. If module interleaving is also to be used in conjunction with the histogramming, then CSR#1<03:00> is loaded with the bit mask identifying which of the bits of CSR#3 are interpreted for subdivision selection. Logical addressing is then enabled by loading CSR#0<01> = 1.

The user should load CSR#16 = 0 to clear overflow count. The active portion of the memory array should also be cleared to zero prior to enabling the histogram mode. Both actions occur under user control.

Using either geographic or logical addressing, CSR#0<08:06> must be set to the encoded value 1. Care must be exercised in setting these bits in that simply writing the value #40 to CSR#0 does not alter the value of CSR#0<07> if it is already set. Therefore it is better practice to first zero the encoded field by writing #1C00000 and then #40 to CSR#0.

Once initialized, a logical address connection to BSM data space without execution of any data cycles (null operation) is processed as an increment command. At AS(d) the address latched into the NTA register is saved in the increment FIFO (discussed below) and an increment operation queued. A geographical address connection followed by a null operation has no effect. Address connections to CSR space do not generate increment commands.

If a logical address connection to a BSM is attempted at such time as the increment FIFO is full and the histogram mode is enabled, then the BSM asserts WT = 1 until the increment FIFO is no longer full.

Except when the increment FIFO is not empty and the histogram mode is enabled, all data space data cycles operate normally. Otherwise, the increment FIFO has priority for memory access as long as an address to increment remains in the increment FIFO. While emptying the increment FIFO, the BSM asserts $WT = 1$ during data space data cycles.

4.2.4 Circular Buffer Mode

In the circular buffer mode, the BSM stores and fetches logical records (blocks) of variable length. The circular buffer mode is selected by setting $CSR\#0\langle 08:06 \rangle = 2$.

Initialization for the circular buffer mode requires a few more steps than for histogramming. First, geographic addressing is used to load $CSR\#3$ with the logical address. If module interleaving is also to be used in conjunction with the circular buffer mode, then $CSR\#1\langle 03:00 \rangle$ is loaded with the bit mask identifying which of the bits of $CSR\#3$ are interpreted for subdivision selection. Logical addressing is then enabled by loading $CSR\#0\langle 01 \rangle = 1$.

The direction of data flow through the BSM is controlled by $CSR\#0\langle 09 \rangle$. If set = 0, then the flow through the circular buffer is from input on the cable segment through memory to output to the auxiliary connector. If set = 1, then the flow is from input on the auxiliary connector through memory to output on the crate segment.

At initialization time both the read and write pointers for the circular buffer, $CSR\#11$ and $CSR\#12$, respectively, are set to zero. The circular buffer alert bit, $CSR\#0\langle 10 \rangle$, is set = 0 at this time. If the circular buffer alert level indicator reported by $CSR\#0\langle 10 \rangle$ is to be interpreted, then $CSR\#13$ is loaded with the occupancy alert limit.

Finally, either by geographic or logical addressing $CSR\#0\langle 08:06 \rangle$ is set to the encoded value 2. Care must be exercised in setting these bits because simply writing the value #80 to $CSR\#0$ does not alter the value of $CSR\#0\langle 06 \rangle$ if it is already set. Therefore it is better practice to first zero the encoded field by writing #1C00000 and then #80 to $CSR\#0$.

Address cycles to either CSR or data space are insensitive to the circular buffer mode of the BSM. Similarly, data cycles in most of CSR space are also insensitive to the circular buffer mode. Only $CSR\#10$, $CSR\#11$, and $CSR\#12$ operate with special characteristics in this mode. For the destination port as designated by $CSR\#0\langle 09 \rangle$, full access to data space is allowed. For the source port any attempted access to data space causes the BSM to generate $SS = 6$ and no data is accepted or asserted.

For random read cycles from the destination port, $CSR\#10$ asserts data from the circular buffer and $CSR\#11$ acts as a address pointer to the memory in data space if the circular buffer mode is enabled. A word is transferred regardless of whether $CSR\#11$ points to data or a control word. For random write cycles from the destination port, the BSM generates $SS = 6$ and accepts no data.

For random write cycles from the source port, $CSR\#10$ accepts data for the circular buffer and $CSR\#12$ acts as a address pointer to the memory in data space if the circular buffer

mode is enabled. A word is transferred regardless of whether CSR#12 points to data or a control word. For random read cycles from the source port, the BSM generates SS = 6 and asserts no data.

For block read cycles from the destination port, CSR#10 asserts data from the circular buffer and CSR#11 acts as an address pointer to the memory in data space if the circular buffer mode is enabled. Block data and control words are read from the memory array and the contents of CSR#11 is incremented circularly until the block length plus the control word count is reached. If the circular buffer is empty before the block is transferred, then the BSM generates SS = 2 and the contents of CSR#11 remains unchanged. For block write cycles from the destination port, the BSM generates SS = 6 and accepts no data.

For block write cycles from the source port, CSR#10 accepts data for the circular buffer and CSR#12 acts as an address pointer to the memory in data space if the circular buffer mode is enabled. Block data and control words are written to the memory array and the contents of CSR#12 is incremented circularly until the block length plus the control word count is reached. If free memory is exhausted before the block is transferred, then the BSM generates SS = 2 and the contents of CSR#12 restored to the value existing prior to the block write. For block read cycles from the source port, the BSM generates SS = 6 and asserts no data.

Random read data cycles to CSR#11 operate normally and assert the current circular buffer read pointer. Random write data cycles from the destination port as designated by CSR#0<09> operate normally and accept the current circular buffer read pointer. However, if the circular buffer mode is enabled and if any CSR write data cycle from the source port as designated by CSR#0<09> attempts to address CSR#11, then the BSM issues SS = 6 and accepts no data. For CSR#11 all block transactions are invalid and generate SS = 6.

Random read data cycles to CSR#12 operate normally and assert the current circular buffer write pointer. Random write data cycles from the source port as designated by CSR#0<09> operate normally and accept the current circular buffer write pointer. However, if the circular buffer mode is enabled and if any CSR write data cycle from the destination port as designated by CSR#0<09> attempts to modify CSR#12, then the BSM issues SS = 6 and accepts no data. For CSR#12 all block transactions are invalid and generate SS = 6.

The BSM implements CSR#13 as an user definable alert level for the circular buffer occupancy. Should the number of words used within the circular buffer exceed the value stored in CSR#13, then CSR#0<10> is set = 1. Otherwise, CSR#0<10> is set = 0.

4.3 Internal Structure Organization

In CSR space, the BSM implements CSR#0, CSR#1, CSR#3, CSR#10, CSR#11, CSR#12, CSR#13, CSR#14, CSR#15, CSR#16, CSR#70, and CSR#71. In data space the BSM includes a hidden increment FIFO, and the memory array.

4.3.1 CSR#0

The implementation of CSR#0 is largely defined by the FASTBUS specification. Table 1 separately identifies functions for each bit within CSR#0 for the read and write operation.

Table 1. Functions for bits in CSR#0.

Bit	Read	Write
00	Error flag	Error flag
01	Logical address enabled	Enable logical addressing
02	-	-
03	BSM allocated	Allocate BSM
04-05	-	-
06-07	BSM mode	Set BSM mode bit = 0 for random access memory = 1 for histogramming memory = 2 for circular buffer = 3 spare
08	Histogram FIFO not empty	-
09	Circular buffer direction	Set direction for aux to crate
10	Circular buffer alert level	-
11	Port priority	Give priority to auxiliary port
12	Double bit error flag	Set double bit error flag
13	Single bit error flag	Set single bit error flag
14	Parity error	Set parity error
15	Active from FSI	-
16	BSM ID	Clear error flag
17	BSM ID	Disable logical addressing
18	BSM ID	-
19	BSM ID	Deallocate BSM
20-21	BSM ID	-
22-24	BSM ID	-
25	BSM ID	Set direction for crate to aux
26	BSM ID	-
27	BSM ID	Give priority to crate port
28	BSM ID	Reset double bit error flag
29	BSM ID	Reset single bit error flag
30	BSM ID	Reset
31	BSM ID	Clear

4.3.2 CSR#1 BSM Address Mask

CSR#1<03:00> defines the bit field included within CSR#3 that distinguishes between subdivisions of a single "logical" BSM module. If assigned the value zero, then module interleaving is disabled. If CSR#1 is not zero, it is the number of subdivisions of the "logical" BSM module minus one and belongs to the set of integers of the form $2^n - 1$

where n is an integer and 2^n is the number of interleaved BSM modules. Figure 4 illustrates the format for CSR#1.

Fig. 4. Format for CSR#1.

This format limits "logical" BSM configurations to those with subdivisions numbering an integer power of two. However, nothing precludes organizing, for example, seven BSM modules as a "logical" BSM with 4 subdivisions plus another "logical" BSM with 2 subdivisions plus a single BSM. In this case the performance varies with the multiplicity of the interleave for the region of memory accessed.

4.3.3 CSR#3 Logical Address Register

In addition to the DA field required by IEEE 960, CSR#3 contains a bit field that distinguishes between subdivisions of a single "logical" BSM module. This field is defined to be the BSM Address (BA) field, and is unique for each subdivision. The width of the BA field is specified by the mask loaded into CSR#1<03:00> by the host processor. Figure 5 illustrates the format for CSR#3.

Fig. 5. Format for CSR#3.

If interleaving is enabled, the BA field can not be modified. Attempted write access gives SS = 6. If interleaving is enabled, the BA field is read as zeroes.

4.3.4 CSR#10 Circular Buffer Contents

Block read and write operations in CSR space are valid for only CSR#10. During these operations CSR#10 operates like a FIFO accepting and asserting data without advancing the NTA register. As discussed elsewhere, CSR#11 and CSR#12 increment at DK(t) for each data cycle for such block read and block write cycles, respectively.

For random read cycles from the destination port, CSR#10 asserts data from the circular buffer and CSR#11 acts as an address pointer to the memory in data space if the circular buffer mode is enabled. A word is transferred regardless of whether CSR#11 points to data or a control word. For random write cycles for CSR#10 from the destination port, the BSM generates SS = 6 and accepts no data.

For random write cycles from the source port, CSR#10 accepts data for the circular buffer and CSR#12 acts as an address pointer to the memory in data space if the circular buffer mode is enabled. A word is transferred regardless of whether CSR#12 points to data or a control word. Random read cycles of CSR#10 from the source port is permitted.

For block read cycles from the destination port, CSR#10 asserts data from the circular buffer and CSR#11 acts as an address pointer to the memory in data space if the circular buffer mode is enabled. Block data and control words are read from the memory array and the content of CSR#11 is incremented circularly until the block length plus the control word count is reached. If the circular buffer is empty before the block is transferred, then the BSM generates SS = 2 and the content of CSR#11 remains unchanged. For block write cycles for CSR#10 from the destination port, the BSM generates SS = 6 and accepts no data.

For block write cycles from the source port, CSR#10 accepts data for the circular buffer and CSR#12 acts as an address pointer to the memory in data space if the circular buffer mode is enabled. Block data and control words are written to the memory array and the content of CSR#12 is incremented circularly until the block length plus the control word count is reached. If free memory is exhausted before the block is transferred, then the BSM generates SS = 2 and the content of CSR#12 is restored to the value existing prior to the block write. For block read cycles of CSR#10 from the source port, the BSM generates SS = 6 and asserts no data.

4.3.5 CSR#11 Read Pointer

In the circular buffer mode, CSR#11 points to the address in the memory array to which the next random read or block read cycle of CSR#10 from the data destination port as designated by CSR#0<09> is to be read. In effect it acts as a variant of the familiar NTA. To protect the read pointer from inadvertent corruption, only the destination port may modify it. This permits more flexible readout algorithms from the destination port.

Random read data cycles to CSR#11 operate normally and fetch the current circular buffer read pointer. Random write data cycles from the destination port as designated by CSR#0<09> operate normally and define a new circular buffer read pointer.

If the circular buffer mode is enabled and if any CSR write data cycle from the source port as designated by CSR#0<09> attempts to address CSR#11, then the BSM issues SS = 6 and accepts no data. For CSR#11 all block transactions are invalid and generate SS = 6.

4.3.6 CSR#12 Write Pointer

In the circular buffer mode, CSR#12 points to the address in the memory array to which the next random write or block write cycle for CSR#10 from the data source port as designated by CSR#0<09> is to be written. In effect it acts as a variant of the familiar NTA. To protect the write pointer from inadvertent corruption, only the source port modifies it.

Random read data cycles to CSR#12 operate normally and fetch the current circular buffer write pointer. Random write data cycles from the source port as designated by CSR#0<09> operate normally and define a new circular buffer write pointer.

If the circular buffer mode is enabled and if any CSR write data cycle from the destination port as designated by CSR#0<09> attempts to modify CSR#12, then the BSM issues SS = 6 and accepts no data. For CSR#12 all block transactions are invalid and generate SS = 6.

4.3.7 CSR#13 Alert Level

The BSM implements CSR#13 as an user definable alert level for the circular buffer occupancy. Should the number of words used within the circular buffer exceed the value stored in CSR#13, then CSR#0<10> is set = 1. Otherwise, CSR#0<10> is set = 0.

4.3.8 CSR#14 Memory Error Summary

The BSM implements CSR#14 to report the syndrome bits associated with the last error. Analysis of the contents of CSR#14 may allow identification of the failed bit in the full memory word (39 bits).

4.3.9 CSR#15 Memory Error Address

The BSM implements CSR#15 to report the address of the last error. Analysis of the contents of CSR#14/CSR#15 may allow identification of the failed memory chip.

4.3.10 CSR#16 Overflow Counter

The overflow counter is used in conjunction with the histogramming mode. It counts the number of times that the incrementer overflows out of the storage word available within the memory array. The overflow counter must be reset by the user prior to enabling the histogram mode.

4.3.11 CSR#70/71 Control for Ports

Each port addresses the port control register as CSR#70. The BSM ports add a port dependent offset to specify the actual address. The crate and auxiliary port offsets are 0 and 1, respectively. Only bit 00 is defined for this register.

If CSR#70<00> = 1, then the crate port has exclusive access to memory until such time as the bit is reset. Similarly, if CSR#71<00> = 1, then the auxiliary port has exclusive access to memory until such time as the bit is reset. If any port has exclusive access to memory, an attempt to set bit 00 = 1 gives response SS = 6.

4.3.12 Increment FIFO

The increment FIFO is only used when the histogramming access mode is enabled. During increment null operations from the crate port or the cable port, the increment FIFO receives addresses from the NTA, buffers addresses to minimize the address connect time, and presents addresses to the increment logic at rates limited by memory speeds.

The increment FIFO is eight words deep to ensure that less than 0.????% of increment transactions generate wait assuming a four-way interleave and an increment transaction rate of 10 MHz. The increment FIFO is 27 bits wide to allow addressing of on-board memory arrays of at least 128 Mwords per module.

4.3.13 Memory array

The memory array is used with all access modes and may be interleaved or not as required by the user. The dual-port provides access to the memory and arbitrates any contention. In the current implementation, the memory array is 8 Mwords deep. The memory width matches the width of a FASTBUS word (32 bits). In addition each word includes extra bits (7) to provide for correction of single bit errors and detection of double bit errors.

Two interleaved BSM modules perform block transfers operations at a rate of 10 MHz. A single BSM module executes block transfers at a rates of 5 MHz.

When CSR#0<08:06> = 0, 1 or 2, the BSM module employs error detection and correction (EDC) techniques to correct all single bit errors and detect double bit errors. Any error found is reported in CSR#14/CSR#15 and indicated in CSR#0, bit 13 or 14 for single or double bit errors, respectively.

5. Software for Sample Applications

5.1 Single module software operation.

Table 2 lists the initialization sequence required to prepare the BSM for operation in the circular buffer mode as in the example of Section 3.1. The sequence assumes that the BSM resides in slot 7 and requires a logical address of 0100_0000h. Also listed is a command sequence to push a block of information into the circular buffer. This is followed by a sequence to pull a block from the buffer. The push sequence would necessarily be executed by the protocol converter and the pull sequence by a master located somewhere on the crate segment.

The crate initialization sequence first arbitrates for bus mastership, establishes a connection with the BSM module, and loads CSR3 with the logical address for the module. Next the master defines the interleave register to specify no interleave for this example configuration. CSR0 is then set to enable logical addressing while at the same time clearing all mode bits. Subsequently the mode bits are set for the circular buffer mode and the direction for circular buffer data flow is set for auxiliary to crate port. Last, the read and write pointers for circular buffer access, CSR11 and CSR12, respectively, are cleared to zero. The crate master then disconnects from the BSM. The module is ready to use.

Access from the auxiliary port side requires no initialization. No arbitration is necessary for access since the protocol converter does not share the auxiliary connector as a bus. The protocol converter simply generates an address cycle to a arbitrary slot supplied by the converter itself, 0 in this example, and initiates circular buffer access via a secondary address cycle selecting CSR10. Logical addressing is not used from this port in this application. The address cycle serves to delineate blocks of data from the optical link. Next, the block write fills a record in memory, which terminates when the address connection is dropped to signal the end of information.

Access from the crate port is essentially conventional. After arbitration, a logical address cycle selects the BSM module and a secondary address cycle selects circular buffer access. A block read transfers a record of information previously deposited in the circular buffer, and an address disconnect terminates the transaction.

Table 2. FASTBUS sequence of transactions for BSM initialization for use with a serial protocol converter and the circular buffer mode.

: following commands are driven from the
: crate segment

ARBITRATION : get bus mastership
PROTOCOL : OFF
PRIORITIZED : OFF

PRIMARY ADDRESS : geographic address
: to CSR space
MODE : (1) SPECIFIC, CSR
ADDRESS : 0000_0007H
PARITY : OFF
ENABLE EG : OFF

SECONDARY ADDRESS
: logical address register
ADDRESS : 0000_0003H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : load it
DATA : 0100_0000H
PARITY : OFF

SECONDARY ADDRESS
: interleave register
ADDRESS : 0000_0001H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : no interleave
DATA : 0000_0000H
PARITY : OFF

SECONDARY ADDRESS
: address CSRO
ADDRESS : 0000_0000H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : enable logical addr
: clear mode field
DATA : 001C_0002H
PARITY : OFF

RANDOM WRITE : set circular mode
: and direction
DATA : 0000_0082H

PARITY : OFF

SECONDARY ADDRESS

ADDRESS : read pointer
: 0000_0011H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : clear read pointer

DATA : 0000_0000H
PARITY : OFF

SECONDARY ADDRESS

ADDRESS : write pointer
: 0000_0012H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : clear write pointer

DATA : 0000_0000H
PARITY : OFF

RELEASE AS : done with CSR space

: Following commands are driven from the
: auxiliary port by the protocol converter.

: Arbitration is not required as the protocol
: converter alone is connected to the port.

PRIMARY ADDRESS : geographic address
: to CSR space

MODE : (1) SPECIFIC, CSR
ADDRESS : 0000_0000H
PARITY : OFF
ENABLE EG : OFF

SECONDARY ADDRESS

ADDRESS : access circular buffer
: 0000_0010H
PARITY : OFF
READ/WRITE : WRITE

BLOCK WRITE : write record

BLOCKSIZE : 5
REPEAT COUNT : 1
PARITY : OFF
DATA =

A5A5_A5A5H
FFFF_FFFFH
0000_0000H
0F0F_0F0FH
F0F0_F0F0H

RELEASE AS : done with block

: Master reads record from circular buffer.

ARBITRATION : get bus mastership
PROTOCOL : OFF
PRIORITIZED : OFF

PRIMARY ADDRESS :logical address
MODE : (1) SPECIFIC, CSR
ADDRESS : 0100_0000H
PARITY : OFF
ENABLE EG : OFF

SECONDARY ADDRESS
: access circular buffer
ADDRESS : 0000_0010H
PARITY : OFF
READ/WRITE : WRITE

BLOCK READ : read "long" record
COUNT : 10000

RELEASE AS

RELEASE BUS

5.2 Software operation with four BSMs and a processor module

Table 3 lists the initialization sequence required to prepare the BSM for operation in the random access mode as in the example of Section 3.2,. The sequence assumes that the BSMs reside in slots 7,8,9, and 10; and loads a logical address of 0100_0000h for the BSM set. Only the initialization for the crate segment is shown; the auxiliary port sequence would be identical except that the arbitration for and release of the bus would be omitted, a single address cycle would permanently connect to the BSM with no release, and the geographic addresses would be 0-3 in place of 7-10, respectively.

In a sequence covering all participating modules in the interleave, the crate initialization procedure first arbitrates for bus mastership, establishes a connection with the BSM module, and loads CSR3 with the logical address for the module as well as the subdivision number within the interleave. Next the master defines the interleave register, CSR1, to specify a four-way interleave for this example configuration. CSR0 is then set to enable logical addressing while at the same time clearing all mode bits. Subsequently the mode bits are set for random access. The crate master then disconnects from the BSM, and handles the next module in the interleave until all have been processed.

Initialization for the processor port would be nearly identical. As noted above, arbitration for and release of the auxiliary port can be omitted since the bus is not shared. Once initialized, the processor should execute a single logical address cycle to the shared BSM

memory to establish the connection to the "logical" BSM module. Thereafter, secondary address are sufficient to access all the memory.

No sample software is shown for crate access to the interleaved data area since it is exceedingly straightforward. It is identical with "normal" random and block data transfers. Similarly, the processor interface need only generate secondary address cycles to identify the starting transfer point for either single or multiple word transfers to and from the BSM, followed by the transfers themselves.

Table 3. FASTBUS sequence of transactions for BSM applications with an independent FASTBUS processor module.

: following commands are driven from the
: crate segment

ARBITRATION : get bus mastership
PROTOCOL : OFF
PRIORITIZED : OFF

PRIMARY ADDRESS : geographic address
: to CSR space

MODE : (1) SPECIFIC, CSR
ADDRESS : 0000_0007H
PARITY : OFF
ENABLE EG : OFF

SECONDARY ADDRESS

: logical address register
ADDRESS : 0000_0003H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : load it for unit 0

DATA : 0100_0000H
PARITY : OFF

SECONDARY ADDRESS

: interleave register
ADDRESS : 0000_0001H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : 4-way interleave

DATA : 0000_0003H
PARITY : OFF

SECONDARY ADDRESS

: address CSRO
ADDRESS : 0000_0000H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : enable logical addr
: clear mode field
DATA : 001C_0002H
PARITY : OFF

RELEASE AS : done with CSR space

PRIMARY ADDRESS : geographic address
: to CSR space
MODE : (1) SPECIFIC, CSR
ADDRESS : 0000_0008H
PARITY : OFF
ENABLE EG : OFF

SECONDARY ADDRESS
: logical address register
ADDRESS : 0000_0003H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : load it for unit 1
DATA : 0100_0001H
PARITY : OFF

SECONDARY ADDRESS
: interleave register
ADDRESS : 0000_0001H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : 4-way interleave
DATA : 0000_0003H
PARITY : OFF

SECONDARY ADDRESS
: address CSR0
ADDRESS : 0000_0000H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : enable logical addr
: clear mode field
DATA : 001C_0002H
PARITY : OFF

RELEASE AS : done with CSR space

PRIMARY ADDRESS : geographic address
: to CSR space
MODE : (1) SPECIFIC, CSR
ADDRESS : 0000_0009H
PARITY : OFF
ENABLE EG : OFF

SECONDARY ADDRESS

 : logical address register
ADDRESS : 0000_0003H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : load it for unit 2

DATA : 0100_0002H
PARITY : OFF

SECONDARY ADDRESS

 : interleave register
ADDRESS : 0000_0001H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : 4-way interleave

DATA : 0000_0003H
PARITY : OFF

SECONDARY ADDRESS

 : address CSRO
ADDRESS : 0000_0000H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : enable logical addr
 : clear mode field

DATA : 001C_0002H
PARITY : OFF

RELEASE AS : done with CSR space

PRIMARY ADDRESS : geographic address
 : to CSR space

MODE : (1) SPECIFIC, CSR
ADDRESS : 0000_000AH
PARITY : OFF
ENABLE EG : OFF

SECONDARY ADDRESS

 : logical address register
ADDRESS : 0000_0003H
PARITY : OFF
READ/WRITE : WRITE

RANDOM WRITE : load it for unit 3

DATA : 0100_0003H
PARITY : OFF

SECONDARY ADDRESS

 : interleave register
ADDRESS : 0000_0001H
PARITY : OFF

READ/WRITE : WRITE

RANDOM WRITE : 4-way interleave

DATA : 0000_0003H

PARITY : OFF

SECONDARY ADDRESS

: address CSR0

ADDRESS : 0000_0000H

PARITY : OFF

READ/WRITE : WRITE

RANDOM WRITE : enable logical addr

: clear mode field

DATA : 001C_0002H

PARITY : OFF

RELEASE AS : done with CSR space

RELEASE BUS

APPENDIX B
Bulk Storage Module Schematics

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B THIRTY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 12/11/90	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE FS CONNECTION SCHEMATIC -SECTION	FILE NO: 99/AN-20182	REV: A	DATE: 12/11/90
---	----------------------------	--------------------------	--------------	--	-------------------------	-----------	-------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B THIRTY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 12/11/98	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE F9 CONNECTION SCHEMATIC -CABLE	FILE NO 99/AM-28184	REV: A	DATE: 12/11/98
---	----------------------------	--------------------------	--------------	---	------------------------	-----------	-------------------

DATE:	1/9/91	REV:	C
FILE NO:	99/JAN-20105		
TITLE: FASTBUS BULK STORAGE MODULE			
PAGE:	1/1		
DESIGN DATE:	3/29/90		
DESIGNER:	D. MAQUEN, PE		
SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.			
4911 B TRINITY DR, LOS ALAMOS, NM 87544 USA

The diagram illustrates the internal logic of a Fastbus Bulk Storage Module. Key components include:

- 74ALS74:** Four D-type flip-flops used for stateful logic, with inputs for clock (C1), set (S), reset (R), and data (D), and outputs for Q and Q-bar.
- 74ALS00:** NAND gates used for basic logic operations.
- 74ALS04:** Hex inverters for signal inversion.
- 74ALS08:** 3-input AND gates for complex signal combination.
- 74ALS32:** OR gates for signal aggregation.
- 74ALS98:** Monostable multivibrators for pulse generation.
- 74ALS94:** 4-bit binary counters for timing or sequencing.
- CS971.0:** A PAL (Programmable Array Logic) component with 12 inputs and 12 outputs, used for custom logic implementation.

The circuit is controlled by several signals:

- POWER_UP:** Indicates the system is powered on.
- LOAD_CS971:** Load signal for the CS971.0 PAL.
- LOAD_CS978:** Load signal for the CS978.0 PAL.
- PORT_BUSY:** Signal indicating the port is busy.
- CABLE_PRIORITY:** Signal indicating cable priority.
- SEL_CYCLE:** Selection cycle signal.
- SEL_CYCLE:** Selection cycle signal.
- SEL_CYCLE:** Selection cycle signal.

The diagram also shows the internal structure of the CS971.0 and CS978.0 PALs, including their input/output pins and internal logic blocks.

A note at the bottom of the diagram states: "REPLACE THIS WITH PART OF A PAL POSSIBLY A 16K4".

[illegible]

The diagram illustrates the internal architecture of the 22448B device, showing the connection between various internal components and external pins. The device is divided into two main functional blocks, each with its own set of pins and internal logic.

Top Block (Pins 1-23):

- DATA, NV, and Control Signals:** Pins 1-23 are connected to internal logic blocks. Key signals include DATA, NV, TFE, ACT, WEN, and various control signals like FREQ, SEL, and ACT.
- Internal Logic:** The top block contains several internal logic blocks, including a 22448B device, which is connected to various pins and internal components.

Bottom Block (Pins 24-41):

- GATE, MODE, and Control Signals:** Pins 24-41 are connected to internal logic blocks. Key signals include GATE, MODE, FREQ, SEL, and ACT.
- Internal Logic:** The bottom block contains several internal logic blocks, including a 22448B device, which is connected to various pins and internal components.

The diagram shows the complex internal wiring and logic that enable the 22448B device to perform its functions, with various signals and control lines connecting the internal components to the external pins.

7	7	4	3	2	1
---	---	---	---	---	---

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: M4 D. MACHEN, PE	DESIGN DATE: 5/25/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE	FILE NO: SS/IN-28822	REV: E	DATE: 10/12/98
--	-------------------------------	----------------------	-----------	------------------------------------	----------------------	--------	----------------

NOTE: THE MODULE MUST BE SET BY JAMPER FOR THE INTERLEAVE EXPECTED.

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 5491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 8/8/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE ADDRESS STRIPING	FILE NO: 99/34-20038	REV: C	DATE: 8/29/90
--	----------------------------	------------------------	--------------	--	-------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B THIRTY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACDON, PE	DESIGN DATE: 8/18/89	PAGE: 1/	TITLE: FASTBUS BULK STORAGE MODULE FFO & CONTROL	FILE NO: 55/JN-20020	REV: E	DATE: 8/25/90
---	----------------------------	-------------------------	-------------	--	-------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B THIRTY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACKEN, PE	DESIGN DATE: 8/18/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE	FILE NO: SSJ/M-20000	REV: C	DATE: 8/29/90
---	----------------------------	-------------------------	--------------	------------------------------------	-------------------------	-----------	------------------

DESIGNER: D. MACKEN, PE	DESIGN DATE: 11/28/89	PAGE: 4/	TITLE: FASTBUS BULK STORAGE MODULE	FILE NO: 99/IN-28873	REV: K	DATE: 3/11/91
SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.	5491 B THIRTY DR., LOS ALAMOS, NM 87544 USA

CAN'T MESS WITH THIS PRINT OR SON HEADLAMP!!

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACKEN, PE	DESIGN DATE: 8/29/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE NEW INCREMENT COUNTER-CSR18	FILE NO: SS/JN-20000	REV: C	DATE: 5/25/98
--	----------------------------	-------------------------	--------------	---	-------------------------	-----------	------------------

DESIGNER: SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGN DATE: 5/29/68	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE DI BUS FULL DOWNING	FILE NO: 591/1A-28165	REV: B	DATE: 1/9/91
---	-------------------------	--------------	---	--------------------------	-----------	-----------------

USE WW PINS ON THE DELAY OUTPUTS FOR FINE TUNING
BRING ALL TAPS OUT.

1. GATES CHANGED TO 74AG-3/13/90

2. Refresh oscillator must be replaced with a DAP oscillator for fab.

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B THIRTY ONE, LOS ALAMOS, NM 87544 USA	DESIGNER: R. Downing	DESIGN DATE: 28 JUN 1989	PAGE: 1/4	TITLE: FASTBUS BULK STORAGE MODULE ABSTRACTION AND NAME	FILE NO: 95/N-28891	REV: C	DATE: 9/13/90
---	-------------------------	-----------------------------	--------------	--	------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: P. Downing	DESIGN DATE: 28-Jun-99	PAGE: 2/4	TITLE: FASTBUS BULK STORAGE MODULE SIGNAL GENERATION LOGIC	FILE NO: SS/JN-20092	REV: D	DATE: 9/18/98
--	-------------------------	---------------------------	--------------	--	-------------------------	-----------	------------------

from ECC
bit error from

DESIGNER:	DESIGN DATE:	PAGE:	TITLE:	FILE NO:	REV:	DATE:
SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.	20 JUNE 1989	4/4	FASTBUS BULK STORAGE MODULE	99/JA-20004	B	12 SEPT 89
3491 B THIRTY DR., LOS ALAMOS, NM 87544 USA			MULTI-ERROR COUNT LOGIC			

MODE_1	MODE_0	OPERATION
0	0	Refresh without scrubbing
0	1	Refresh with scrubbing/refresh.
0	0	All FRSs driven with RAS
1	0	Only one CSBn driven with CSB.
1	0	Read/write normal mode.
1	1	Clear refresh counters on RAS.

when doing
ARIMA, the model
is $00 = \text{ARIMA}$

41. *unabridged*
(unabridged)

1. At power up:
 - a) allow memory chips 200µsec to stabilize before cycling.
 - b) run a minimum of 8 refresh cycles after 200µsec delay.
2. Initialize memory before using to eliminate read ecc errors.

REMOVED GENERIC GATES FOR SIMULATION-3/13/90
LEAVE ALL PARTS. 12/10/90

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: R. DOMING	DESIGN DATE: 9/29/89	PAGE: 2/2	TITLE: FASTBUS BULK STORAGE MODULE FAB INT LOGIC	FILE NO: 559/IN-28898	REV: C	DATE: 3/13/90
--	------------------------	-------------------------	-----------	---	--------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: R. DOMANIG	DESIGN DATE: 29 JUNE 1989	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE	FILE NO SSI/A-20097	REV C	DATE: 24 OCT. 1989
--	-------------------------	------------------------------	--------------	------------------------------------	------------------------	----------	-----------------------

8	7	6	5	4	3	2	1
---	---	---	---	---	---	---	---

8	7	6	5	4	3	2	1
---	---	---	---	---	---	---	---

1-2
Split
2/23/80
Self-procedure

SS/JV-2005
 CHECK BIT MEM 2.DRW
 RWD 6/28/89

1-E
 SPLIT
 2/12/90
 SFF overflow

C HIGH DATA ON D TO D
C LOW LATCH DATA ON D

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B THIRTY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. McNamee, PE	DESIGN DATE: 8/9/98	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE TRANSPARENT LATCH	FILE NO: SS/JM-LATCH	REV: A	DATE: 8/9/98
---	-----------------------------	------------------------	--------------	---	-------------------------	-----------	-----------------

541 => 74ACT 16204

3x30 9-1

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MAOREN, PE	DESIGN DATE: 5/24/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE MULTIPLIER - 8 BIT	FILE NO. SSR/SP-2/894	REV: C	DATE: 6/79/89
--	----------------------------	-------------------------	--------------	---	--------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 5/24/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE MULTIPLEXER - 24 BIT	FILE NO. SS/IN-21001	REV. D	DATE: 8/29/98
--	----------------------------	-------------------------	--------------	---	-------------------------	-----------	------------------

Note: USE 74ALSCT

0000 0000 0000

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.
3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER: MARY HARRY

DESIGN DATE: 5/25/80

PAGE: 1/1

TITLE: FASTBUS BULK STORAGE MODULE

BUFER-24 BIT

BUF-24

FILE NO: 981/24-71002

REV: A

DATE: 5/25/80

CONC. OPT. 111 C

Ph BUF_32

NOTE: USE 74LSCT

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: MARY HARDY	DESIGN DATE: 5/25/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE BUFFER - 32 BIT BUF_32	FILE NO: 991/JN-21004/3	REV: A	DATE: 5/25/89
--	-------------------------	-------------------------	--------------	--	----------------------------	-----------	------------------

SCENIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: DRN MARY HARDY	DESIGN DATE: 5/21/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE MULTIPLEXER - 8 BIT	FILE NO: 88/JN-21004	REV: C	DATE: 8/29/98
--	-----------------------------	----------------------	-----------	---	----------------------	--------	---------------

MUX BUF_24	
*D1	Y
*D2	Y
*C	O
*WS	O
*CLK	

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.	DESIGNER: DPM	DESIGN DATE: 5/24/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE	FILE NO: 59/IN-2100/5	REV: 0	DATE: 8/29/98
5491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: MARY HARDY

CONFIDENTIAL

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B THIRTY DR., LOS ALAMOS, NM 87544 USA	DESIGNED: D. MACHEN, PE	DESIGN DATE: 9/5/90	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE 24-BIT REGISTER	FILE NO: SS/M-21000	REV: A	DATE: 9/5/90
---	----------------------------	------------------------	--------------	--	------------------------	-----------	-----------------

APPENDIX C
FASTBUS Slave Interface Specifications

1 The FASTBUS Slave Interface

1. Introduction

The FASTBUS Slave Interface (FSI) is a single chip solution for general purpose interfacing of FASTBUS slaves. Built with a Raytheon ECL gate array, the FSI may be connected directly to the FASTBUS. The user interface models the user application as a memory module and provides simple memory-like control signals. Optional functions include advanced MS code operations at each edge of DS, and on-chip support for multiple-module addressing. No master logic functions are supported.

1.1 Standard Functions

The FSI supports all mandatory IEEE 960 slave specifications. The FSI handles geographical, logical, and broadcast addressing using on-chip registers where necessary with the latter two modes. Data cycle support includes single, secondary address, block, and pipeline access. The application independent bits of CSR#0 are also implemented on-chip thus eliminating the requirements for substantial random logic for CSR#0.

1.2 Advanced Modes

To maximize the benefit of shifting the diverse FASTBUS control functions into a single chip, unassigned (but much discussed) MS codes have been fully implemented and may be enabled simply by strapping the appropriate pin. Similarly, strapping another pin enables an advanced addressing technique whereby multiple modules may respond as if individually addressed in turn.

1.3 Configurability

The FSI chip is configurable with option enable pins permitting selection of various features. Separate pins are provided to enable use of CSR#3 and CSR#7 for control of logical addressing and class N broadcast addressing, respectively. Use of the on-chip CSR#0 is separately enabled. With a set of five pins, the user may specify the width of the internal address field for logical addressing. Another option allows the user to select either SS = 1 or WT = 1 responses for internal busy conditions.

2. FSI Implementation

2.1. Package

The FSI is a 232 pin PGA ceramic package with dimensions of 1.8" x 1.8".

2.2. Power Dissipation

The FSI dissipates 7 Watts.

2.3. Connections to FASTBUS

All FASTBUS signals are connected to the FSI except those exclusively reserved for arbitration for and retention of bus mastership. Since the nominal capacitance for the ceramic package is 5.5 pf, the FSI may be connected directly to the segment connector for address/data and protocol signals. If trace lengths become excessively long due to printed circuit board layout factors, buffering may be introduced.

2.4. Timing Controls

Two pairs of signals are used to control the timing response of the FSI. The first pair, RESPONSE_CTL_OUT and RESPONSE_CTL_IN, are used to provide the appropriate delay between the internally generated response signals and AK and DK applied to the segment. For example, this delay may cover actual propagation delays necessary to drive valid SS codes and optional data onto the bus. Should AK require different delays from DK, signal ADDRESS_CYCLE may be used to distinguish between the two, perhaps selecting different taps on the same delay line. RESPONSE_CTL_OUT is asserted until the FSI senses RESPONSE_CTL_IN asserted.

The second pair of timing signals allows the user to signal when data is either asserted or accepted by the slave user-application logic. These signals are only applicable to data cycles and their asynchronous handshake precedes the RESPONSE_CTL_OUT/RESPONSE_CTL_IN handshake discussed above. Each data cycle seen by the attached slave results in the generation of TRANSFER_REQUEST which generally follows the DS line. After an appropriate delay defined by the user, TRANSFER_DONE is toggled matching TRANSFER_REQUEST to signal the FSI that the transfer is complete, and data either accepted or asserted on the buffered data bus (BAD bus) available to the user. The operation of TRANSFER_REQUEST and TRANSFER_DONE is analogous to that of DS and DK, respectively.

Some data cycles do not actually transfer data to or from the user application. For example data accepted or asserted by a CSR register or NTA register on-board the FSI does not require any user response except that TRANSFER_DONE must be toggled in response to TRANSFER_REQUEST. Such data cycles are not

accompanied by assertion of TRANSFER_ENABLE. Those other data cycles delivering or generating user data explicitly have TRANSFER_ENABLE asserted.

2.5. Connections to User Application

Connections to the user application divide into five basic categories. First, there are the memory interface signals. TRANSFER_ENABLE, TRANSFER_REQUEST, and TRANSFER_DONE are in this category. Other memory signals include BLOCK_MODE (indicates multiple word transfer), NTA_ACCESS (transfer of secondary address, which is always transparent to the user application), READ (or not write), CSR_SPACE (or not data space), and NTA(31:0) for the accessed address. NTA_CARRY_IN allows the user to control whether or not the NTA register advances during block mode accesses. The signal NTA_8TO29_NE_0 is asserted when bits 8 through 29 of the NTA register are not equal to zero.

A second category controls the generation of slave status codes. These include BUSY (user application is currently busy), EOB (end of block encountered), NOT_VALID (address or data is not valid), REJECT (data not accepted or not asserted), and SET_SS3 (generate module dependent SS = 3).

The third category bundles together all the mandatory control signals associated with the operation of the FSI that every user-application requires plus often used optional signals for CSR#0. Mandatory signals include signals such as power-up (input) for initialization and module connection status (output). CSR#0 interface signals report CSR#0 states to the user while others act as status or commands driving the state of CSR#0.

The forth category consists of option selection signals. These signals are typically static, and fix many of the operating modes of the FSI. These options include specification of the width of the internal address used during logical address cycles, definition of the registers supported on-board the FSI chip, and the mode for FSI response when the user application is found "busy." Either an SS = 1 or WT assertion is possible for the last case.

The final category is the set of miscellaneous signals not included in any other category. These special purpose signals are infrequently used, and include the signal asserted when a null cycle has been executed (connection followed by disconnect with no intervening data cycles), and the signal asserted when the FSI is operating in a multi-module mode.

2.6. Timing Performance

Tables 1 and 2 summarize the results of simulations of the FSI prior to actual layout of the ASIC. Table 1 gives the propagation delays for changes of various input signals on both the FASTBUS and user application interfaces. Table 2 lists signals that are latched within the FSI by the TRANSFER_DONE(t) transition. Each must satisfy setup and hold conditions as noted.

Table 1. Propagation delays obtained from simulation-ECL array implementation.

Transition	Delay (ns) and test condition
EG -> AK (geographic address)	14
AS -> AK (logical address)	16
AS -> NTA	10
EG -> Response_Control_Out	8
Response_Control_In -> AK	6
BAD -> PA (read)	7
BAD -> AD (read)	4.5
AD -> BAD (write)	2.5
DS -> DK	14
DS -> NTA (write)	9
TRANSFER_DONE -> DK(t)	10
TRANSFER_DONE -> NTA+1	9 (block/pipe)
DS(t) -> TRANSFER_ENABLE	10 (CSR space)
DS(t) -> TRANSFER_ENABLE	6 (data space)
EG -> CSR_SPACE	8
DS(t) -> CSR_SPACE	12
DS(t) -> AD	10 (broadcast TP-pin read)
TRANSFER_DONE -> SS	6

Values in Table 2 vary widely depending on the application signal. The setup time for the NTA_CARRY_IN signal is especially long since the worst case must allow the carry-in to propagate through 32 bits of the NTA incrementer. On the other extreme, the inputs which control the SS-response pass directly to internal latches, and therefore they require no setup time, but demand hold time while the slower control for the latch is generated. Inputs for TP-pin control have intermediate setup values due to delays in the multiplexer that select the input for the TP-pin, delays in decoding the bit to drive on the AD bus, and then driving the bus.

Table 2. Setup and hold times calculated with respect to TRANSFER_DONE(t).

FSI application input	Setup (ns)	Hold (ns)
BAD	1	1
BUSY	0	2
EOB	0	2
NOT_VALID	0	2
REJECT	0	2
SET_SS3	0	2
NTA_CARRY_IN	23	2
DATA_AVAILABLE	5	0
DEVICE_AVAILABLE	5	0

3. Operational Details

3.1. Address Cycles

The FSI supports all IEEE 960 addressing modes. While geographical addressing has no special requirements beyond supplying the geographical address, GA, to the FSI, a logical address must be preceded by initialization of the contents of the on-chip CSR#3, setting the on-chip bit CSR#0<1>, and setting the bit count for the range of the internal address for the module using the encoded bus, LOGICAL_WIDTH. If the encoded value of LOGICAL_WIDTH is W, then the width of the internal address in bits, B, is given by $B = W + 1$. Thus, the internal address may not have width zero. A broadcast address may connect immediately (general broadcast), connect if bus contents are consistent with the previously loaded on-chip CSR#7 (class N broadcast), or conditionally connect subject to the test specified by the TP-pin broadcast mode. For the TP-pin broadcast modes, the FSI supports the optional read data cycle that may immediately follow the address cycle, and the optional write data cycle that may disconnect some or all of the connected modules prior to normal data cycles. Table 3 identifies what signal drives the TP-pin as a function of the broadcast mode, and whether the FSI sources the signal from internal states or from user-application interface signals.

Table 3. Supported broadcast modes and signals driven onto the the TP-pin during optional read-data cycle immediately following the address cycle. The source column identifies whether the FSI drives the TP-pin from an internal register or must pass along input from the user application.

Broadcast case	Signal driven to TP	Source
General broadcast (case 1)	None	
Class n broadcast (case 2)	None	
Case 3	DATA_AVAILABLE	External
Case 3b	DEVICE_AVAILABLE	External
Case 4	Logic level true	Internal
Case 5	SR	Internal
Case 6	SR_FLAG	Internal

The function of the TP-pin is integrated into the FSI. To facilitate use of the FSI with cable segments which have no TP-pins, the FSI multiplexes the TP response, for an optional MS = 0 read cycle immediately following a broadcast address cycle, directly onto the AD bus. The bit selected is determined by the GA inputs. Similarly, TP input for an optional MS = 0 write cycle during the extended broadcast addressing process is taken directly from the the AD bus.

The ATTACHED line provides the user-application connection status information. ATTACHED is set by the FSI after connection to indicate that the module is attached as the current slave.

Independent of addressing mode, the FSI initiates connections by generating CSR_SPACE reflecting the type of space addressed, ADDRESS_CYCLE to indicate that an address cycle is in progress, a 32-bit wide NTA bus, and RESPONSE_CTL_OUT to trigger the user-application timing control. The user logic validates the address cycle, generates any controls for generation of non-zero SS responses, and asserts RESPONSE_CTL_IN to initiate the AK response. Note that for broadcast addresses, RESPONSE_CTL_IN must be asserted as for other address cycles, but the AK is suppressed internally by the FSI. See the timing control section for additional detail concerning required timing delays for RESPONSE_CTL_IN.

Both CSR#3 and CSR#7 are implemented to provide the maximum flexibility for the user application. CSR#3 is a full 32-bit wide register. CSR#7 is only 16 bits wide as required by the IEEE 960 specification. The registers are separately enabled/disabled for internal use on the FSI by strapping CSR3_EN and CSR7_EN.

3.2. Data Cycles

The FSI considers an independent data cycle in progress anytime DS and DK differ. Secondary address cycles are considered data cycles. At DS(t) the data

cycle begins. For write operations, data flows from the AD pins to the BAD pins, and in the opposite direction for read functions. Selected on-chip registers are transparent during write data cycles and assert data throughout read cycles. To reduce susceptibility to noise, the RD signal is latched during data cycles and passed to the user application as READ. Similarly, the MS lines are internally latched and used to generate user-application signals BLOCK_MODE and NTA_ACCESS. BLOCK_MODE is asserted if either block (MS = 1) or pipeline (MS = 3) mode is invoked, and NTA_ACCESS is asserted if secondary address mode (e.g. DS(u) and MS = 2) is referenced.

Timing for off-chip accesses is provided through interface signals TRANSFER_REQUEST, TRANSFER_DONE, and TRANSFER_ENABLE. At DS(t) time, signal TRANSFER_REQUEST is toggled. For all off-chip accesses TRANSFER_ENABLE is also asserted. Upon seeing TRANSFER_REQUEST(t), the user application initiates logic to access either CSR or data space memory. When the data has been accepted or asserted as directed by READ, the user application drives TRANSFER_DONE to match TRANSFER_REQUEST.

If the data cycle accessed on-chip storage, TRANSFER_REQUEST is processed as above, but TRANSFER_ENABLE remains reset throughout the data cycle. The user application remains responsible for generating the TRANSFER_DONE synchronization handshake.

The FSI generates its internal DK(t) upon receipt of TRANSFER_DONE(t). At this time, the protective buffer, which latches the bus contents during cycles, drives the AD bus if a read cycle is in progress. At the same time, the NTA register is advanced if a block or pipeline mode operation is active. Then a RESPONSE_CTL_OUT/RESPONSE_CTL_IN handshake analogous to that performed during the address cycle follows to ensure a valid SS response is available. Should different timing requirements be necessary, ADDRESS_CYCLE is reset to enable discrimination. After RESPONSE_CTL_IN is asserted by the user application, the FSI generates DK(t). The data cycle is ended at this point. The SS response and any read data remains asserted until any subsequent DS(t) or AS(d).

The user application controls whether the NTA advances during block or pipeline mode cycles with the signal NTA_CARRY_IN. If NTA_CARRY_IN is at logic 1, then the NTA advances normally. If at logic 0, the NTA holds its value unchanged as is appropriate for FIFO-like devices.

3.3. Slave Status Response

Slave status (SS) response is controlled by the user application through use of five interface signals. All are applicable for data cycles, while only two control primary address cycles. The two signals common to address and data cycles are BUSY and NOT_VALID. An asserted BUSY signal indicates the user application is unable to complete the requested cycle at the current time, and either WT is set

or SS = 1 generated depending on whether WT_EN is enabled or disabled, respectively. The NOT_VALID signal reflects the validity of the current NTA address. For data cycles, NOT_VALID may also mean that the data is invalid.

Other SS response control signals reserved for data cycles include REJECT, EOB, and SET_SS3. When NOT_VALID is asserted, REJECT set or reset controls issuance of either SS = 6 or SS = 7, respectively. The EOB signal is generated by the user application during block or pipeline transfers when the last word in the block is already transferred and yet another transfer is attempted. The EOB signal is only examined during block or pipeline transfers. The response SS = 3 is generated when SET_SS3 is asserted.

In the event that more than one of the SS response control lines are set at once, some controls have priority over others. Listed from highest priority, the signals are SET_SS3, BUSY, EOB, and NOT_VALID. REJECT is subordinate to NOT_VALID, and has meaning only when NOT_VALID is asserted.

3.4. CSR#0

All user-application-independent bits defined for CSR#0 are supported on board the FSI. This set consists of the error flag, logical address enable, run/halt, module allocated, SR enable, SR flag, parity error on FASTBUS, and module active (bits 0,1,2,3,4,5, 14, and 15, respectively). Different functions are supported for each of the bits as required by the FASTBUS specification.

The error flag read from bit 0 is the logical OR of the user-application signal SET_ERROR_FLAG, the internal latch that may be written to simulate an error, and CSR#0<14>. The latch is cleared by CLEAR_ERROR, which is discussed below.

Bit 1 for control of logical addressing is set/reset according to IEEE 960. The status of bit 1 is reported to the user application with signal LOGICAL_ADDR_EN.

Bit 2 for run/halt control is set/reset according to IEEE 960. The status of bit 2 is reported to the user application with signal RUNNING. If in the run state, the user application may change the status to halt by asserting the SET_HALT line.

Bit 3 for recording module allocation status is set/reset according to IEEE 960. The status of bit 3 is not reported to the user application.

Bit 4 for enabling SR interrupts is set/reset according to IEEE 960. The status of bit 4 is not reported to the user application.

Bit 5 for recording pending SR interrupts is set/reset according to IEEE 960. The status of bit 5 is reported to the user application with signal SR_FLAG. If SR_FLAG is reset, the user application may request service pending by asserting

SET_SR. Upon seeing SR_FLAG asserted, the user application resets SET_SR, and makes no further SR interrupt requests until SR_FLAG is reset. In this manner, no SR interrupts are lost.

Bit 14 for reporting FASTBUS parity errors is set/reset according to IEEE 960. Bit 14 is not available to the user application, but may be accessed by a master with a CSR0 read. During the data cycle that sets bit 14, the signal PARITY_ERROR is asserted, and the user application must decide whether the data is valid and to be accepted, or in error and to be accepted or rejected. The user application is therefore responsible for determining the SS response to parity errors; testing procedures may require that SS = 0 be returned upon detection of a parity error. Thus signals NOT_VALID and REJECT must be defined in response to PARITY_ERROR. However, CSR#0<0,14> are automatically set to indicate the parity error.

Bit 15 passes the status of the ACTIVE line supplied by the user application.

Since significant parts of CSR#0 are defined by the user application, signals LOAD_CSR0 and READ_CSR0 are provided by the FSI to facilitate the application. The user application must assert at least the device ID while READ_CSR0 is asserted, and may also assert any other device-dependent bits. User-application bits corresponding to bits defined in the FSI internally are ignored.

When asserted during LOAD_CSR0 operations, bits 16, 30, and 31 have special meaning. Under these conditions, assertion of bit 16, 30, or 31 generates signal CLEAR_ERROR, RESET, or CLEAR_DATA, respectively. These signals are used internally and are also externally supplied to the user application. Internally, CLEAR_ERROR resets bit 0, simulate error, and bit 14, parity error. Externally, it must clear all sources for errors contributing to SET_ERROR_FLAG.

The scope of RESET and CLEAR_DATA is defined in IEEE 960 Section 8.18. As supplied from the FSI, RESET also includes a term asserted when power is applied to the module, *i.e.*

$RESET = AD\langle 30 \rangle * LOAD_CSR0 + POWER_UP$. In contrast, CLEAR_DATA is derived from $AD\langle 31 \rangle * LOAD_CSR0$ alone.

CSR#0 is also affected by the RESET_BUS signal, which is supplied by the user application when the bus is not halted and and reset bus is asserted. Logically $RESET_BUS = BH \sim \int RB$, where $\int$ denotes that the term is integrated as required by the specification. Both logical addressing enable and run/halt, bits 1 and 2, respectively, are reset when RESET_BUS is present. The RESET_BUS signal immediately disconnects the module thereby ensuring AK and DK are no longer asserted. The internal CSR#0 register is separately enabled/disabled for implementation on the FSI by strapping CSR0_EN.

3.5. Parity Detection/Generation

The FSI always checks parity during address cycles and write data cycles if PE is asserted. On read data cycles, parity is generated except on broadcast operations. To disable parity detection/generation, the PE and PA lines are not connected. During any address cycle, detection of a parity error inhibits a connection. During data write cycles, a parity error results in setting CSR#0<14> as discussed in the preceding section.

3.6. Advanced MS Modes for Data Cycles

Advanced MS modes not specified in IEEE 960 are available to the user application by strapping pin LA_MODE_EN of the FSI. As noted in Section 3.2, the FSI considers an independent data cycle to be in progress during times when DS and DK differ. Thus the slave may execute an independent data cycle initiated by DS(u) and another at DS(d). In other words, the phase of DS and the values of the MS codes specify 16 different data cycle operations. For advanced mode operation, the functions defined for these 16 options are shown in Table 4. Note that IEEE 960 data cycles comprise a compatible subset with the advanced set.

The advanced MS codes have been selected to create a symmetric code set such that all defined operations may be executed on either phase of DS(t). Thus a master exploiting this capability never "wastes" cycles, *i.e.* inserts cleanup cycles only where, on a voltage driven bus, traffic switches from read to write.

Three fundamentally new operations are introduced with the new MS code assignments. First, protective buffer access enables transmission of data last accepted or asserted by the slave. This operation is useful for module debugging and for error recovery should retransmission of data be required due to corrupted bus contents during a read operation. FIFO-like devices no longer require special logic to prevent loss of data for possible reread operations.

The second new operation enables arbitrary selection of the space (either CSR or data) to that selected at primary address time whenever the secondary address register is loaded. For example, this allows initial connection to CSR space, a sequence of control data cycles, then switching to data space without the necessity of terminating and re-establishing the AS-AK lock. The space selection operations simultaneously define the effective addressed space and the NTA. Use is limited to write cycles. Note that the usual secondary address write function (MS = 2,6) is interpreted as select the space attached at primary address time, and load the NTA. The usual secondary address read function simply reads the current NTA, which is unique. A symmetric read operation for the opposite space function code (MS = 7) would not be appropriate as it would select the opposite space and read the NTA but leave it unaltered. MS = 7 read is therefor used for another purpose.

The third new operation allows CSR#0 to be read any time without the necessity of switching to CSR space and setting the NTA to zero. This operation shares the

MS operation code with the opposite space selector function defined above, and the state of the RD line decodes the two options.

Slave status response for the new codes are derived from those already established for the IEEE 960 codes. If enabled for the advanced MS codes, single, secondary address, and cleanup cycles give results (data and SS response) independent of the phase of DS. The new protective buffer cycles return SS = 0 except when a protective buffer write cycle encounters a parity error on FASTBUS.

Table 4. Advanced MS codes for data cycles in the FSI. References to NTA(P/A) denote space switch to that selected at primary address time coupled with load of the NTA register. Reference to NTA(P/A~) is similar except that space selected is opposite to that selected at primary address time. If the advanced MS codes are disabled, then MS codes 4-7 are reserved and return SS = 6 to the master. MS code assignments are identical with those implemented by R. Skegg at TRIUMF for the protocol control logic (PCL).

MS Code	DS(u)	DS(d)
0	Single word	Cleanup
1	Block	Block
2	Load NTA(P/A) for RD = 0 Read NTA for RD = 1	Cleanup
3	Pipeline	Pipeline
4	Cleanup	Single word
5	Protective buffer	Protective buffer
6	Cleanup	Load NTA(P/A) for RD = 0 Read NTA for RD = 1
7	Load NTA(P/A~) for RD = 0 Read CSR#0 for RD = 1	Load NTA(P/A~) for RD = 0 Read CSR#0 for RD = 1

3.7. Advanced Multi-Module Addressing

Often it would be useful to have more than one FASTBUS module behave in a collective way such that the group acts as a single larger FASTBUS module. To clarify the concept, consider two examples. First, suppose that a memory module with large capacity is required in your application. The amount of memory supported by a single module is not sufficient for your task, but two modules would do the job. Unfortunately, your software assumes that all the memory is contained within a single module so that block transfers that cross the address boundary between the two modules will cause SS = 2 errors. You want to bind the two memory modules so that when you cross from the address space of the first to the address space of the second, no SS = 2 is reported and no additional primary address cycle is required. The FSI interleave capability provides this option. After running the two modules in this mode, you decide that you could improve your memory performance if you merged the two

memories such that all even addresses were seen in the first module and the odd addresses in the second module. This second example is also available with the FSI interleave option.

The FSI chip supports the multi-module connection during primary address cycles with logical addresses and subsequent data cycle access through use of interleaved address space. Two basic concepts must be established at this time, multi-module connections and interleaved address space. First, we define the concept of multi-module connections. The FSI implements CSR#1 and CSR#3 as on-chip 32-bit wide registers. Bits defined by CSR#1 within the range of the internal address field for the module are used to determine how many cooperating modules are connected at address time and may selected during subsequent data cycles. The multiplicity of the interleave may be determined by first counting the bits of CSR#1 that are set to 1 within the range corresponding to the internal address field. Then for bit count, N , the multiplicity, M , is given by $M = 2^N$. At primary address time, all M modules connect as slaves, but only one will respond at a time. The M modules are differentiated by unique bit patterns in CSR#3 using those bit positions identified by CSR#1. At primary address time, the cooperating module or *subdivision* with bits of the internal address matching the corresponding bits of CSR#3 for bit positions identified by CSR#1 is initially identified as the selected subdivision, *i.e.* the one that will provide the SS response.

The action of data cycles directed to data space may alter the subdivision identified as selected. During cycles, the NTA may be loaded or incremented. The resulting NTA is compared with CSR#3 for those bits identified by CSR#1, and a new subdivision is selected. Single, cleanup, and protective buffer access do not change the NTA, and hence can not change the selected subdivision. Neither can any CSR data cycle alter the selected subdivision.

For correct operation all subdivisions maintain essentially identical internal control structures. Each NTA must be identical throughout the set of cooperating modules. To maintain coherency of the CSR registers, a CSR write is processed by all subdivisions (like a broadcast to each of the subdivisions) with the selected subdivision providing the slave status response. Caution must be exercised to ensure any asymmetry established during initialization, *e.g.* unique contents for CSR#3, is not inadvertently destroyed with subsequent CSR write cycles.

For multi-module data cycles, the signal TRANSFER_ENABLE is generally set only within the selected subdivision. The FSI holds TRANSFER_ENABLE reset for all other members of the cooperating modules unless a write to off-chip CSR space is in progress.

To clarify the concept of interleaved address space, we return to the memory module examples introduced above. Suppose each of the two modules has an

internal address space of 64K. To achieve a two-way interleave, we must set one bit in CSR#1. The position of this bit will determine how the combined address space of the two modules is mingled. For the first example where the address space of one flows into the address space of the other, the contents of CSR#1 must be 0001_0000 in hexadecimal. This indicates that bit 16 determines which subdivision is selected. For the second example, CSR#1 must have the value 0000_0001 which indicates that bit 0 selects the subdivision.

Let us continue to explore some FASTBUS operations that are possible in the second example. With the 16-bit internal address space per module and the two-way interleave, the effective internal address is 17 bits leaving 15 bits for the logical address. Suppose that the logical address of the module pair is 1234_0000 so that CSR#3 contents for the module with even memory addresses is 1234_0000 and 1234_0001 for odd addresses. Further assume that the NTA is cleared to zero, and that we wish to execute a data space block read of length 5. Initially, the subdivision with CSR#3 = 1234_0000 is selected. As the block read starts, data is first read from this module. After incrementing the NTA to 0000_0001, bit 0 identified by CSR#1 changes in the NTA, and selects the subdivision with CSR#3 = 1234_0001 as the next source for data. Similarly, after incrementing again to 0000_0002, the subdivision with CSR#3 = 1234_0000 is then selected since bit 0 of the NTA matches bit 0 of CSR#3. The data source alternates until five reads are completed leaving the NTA with the value 0000_0006. At this point if a secondary address write is executed, the module with CSR#3 = 1234_0000 is selected and provides the SS response. If the new NTA is even, then the module remains selected until subsequent data cycles alter the contents of the NTA bit 0 as identified by CSR#1, or until both subdivisions are disconnected at AS(d). The example shows that data space is interleaved on a single word basis between two modules.

Returning to the first example, we consider another block read that crosses the boundary between the two modules. For this example CSR#1 has already been defined as 0001_0000, and CSR#3 is 1234_0000 in the module with the lower memory block and 1235_0000 for the upper memory block. Further assume that the NTA is set to 0000_FFF0, and that we wish to execute a data space block read of length 30. Initially, the subdivision with CSR#3 = 1234_0000 is selected. As the block read proceeds, 16 data words are first read from this module since bit 16 of the NTA matches bit 16 of CSR#3. After incrementing the NTA to 0001_0000, bit 16 identified by CSR#1 changes in the NTA, and selects the subdivision with CSR#3 = 1235_0000 as the next source for data. After crossing the module boundary, the subdivision with CSR#3 = 1235_0000 remains selected since bit 16 of the NTA does not change as the NTA advances to its final value of 0001_000F. The example shows that data space is interleaved on a 64K block basis between two modules.

The concept of interleaved address space is very general and limited only by practical considerations such as segment size and internal module addressing

capacity. Note that the FSI does not remove any interleave bits identified in CSR#1 before this address is presented to the user application. While this would be desirable, generalized shift and select logic does not fit in the semi-custom devices available today. The appropriate shift and bit selection functions (NTA address formatting) are left for implementation in the user application.

When addressed by either a geographical or broadcast address cycle, the FSI inhibits interleaving. All data space is then sequentially addressed.

Interleaving may be enabled by strapping the CSR1_EN pin. When disabled, the internal CSR#1 of the FSI is inoperative, and an external alternative may be implemented like any other application-specific CSR register.

3.8. Bus Enable Signals

Two user application signals control output of the BAD and NTA buses, OE_BAD and OE_NTA, respectively. If reset, these controls hold their respective bus low. If OE_NTA is set, the NTA bus is driven. If OE_BAD is set, the BAD bus is driven except when a FASTBUS read is in progress.

3.9. Other Operations

The FSI supplies the user application with two signals sometimes useful in special applications. The FSI generates the signal INTRLEAVE_ACTIVE whenever the module has been logically addressed and the CSR1_EN is strapped to enable CSR#1 operation. The INTRLEAVE_ACTIVE signal is particularly useful for enabling a NTA address formatter in the user application. This formatter discards address bits from the NTA address that CSR1 indicates are used for subdivision selection. NULL_CYCLE is generated when a disconnect immediately follows a connect with no intervening data cycles. If module interleaving is enabled, then only the selected subdivision can generate NULL_CYCLE.

Appendix A - Alphabetical listing of the pinout signals referenced by the FSI

ACTIVE	Status bit 15 of CSR#0 supplied by the user application. Used to indicate a busy condition.
AD(31:0)	FASTBUS address/data.
AK	FASTBUS protocol.
AS	FASTBUS protocol.
ATTACHED	Status for slave attachment. Asserted by FSI to indicate subdivision is attached.
BAD(31:0)	Buffered data bus.
BLOCK_MODE	Asserted by FSI for either block or pipeline mode active.
BUSY	Asserted by user to indicate that an address or data request has been received while the subdivision is busy.
CLEAR_DATA	Asserted by FSI during CSR#0 load if AD31 is asserted.
CLEAR_ERROR	Asserted by FSI during CSR#0 load if AD16 is asserted.
CSR0_EN	User signal indicating that CSR#0 bits 0 to 5, bit 14, and bit 15 are supported on-board the FSI chip.
CSR1_EN	User signal indicating that CSR#1 internal to the FSI chip is used and that interleave mode is possible.
CSR3_EN	User signal indicating that CSR#3 internal to the FSI chip is used and that logical addressing is possible.
CSR7_EN	User signal indicating that CSR#7 internal to the FSI chip is used and that case 2 (class N) broadcast addressing is possible.
DATA_AVAILABLE	User signal used for the TP-pin signal during the optional read following a case 3 broadcast primary address cycle. Indicates module contains data to read.
DEVICE_AVAILABLE	User signal used for the TP-pin signal during the optional read following a case 3a broadcast primary address cycle. Indicates module is available for operation.
DK	FASTBUS protocol.
DS	FASTBUS protocol.
EG	FASTBUS protocol.
EOB	Asserted by user after passing end of block for block transfer or pipeline operation.
EXT_LOAD_NTA	Reserved. Asserted by user when contents of user driven BAD bus are to replace the contents of the NTA register within the FSI.
EXT_NTA	Reserved. Asserted by user when the contents of the NTA register have been user initialized.

GA(4:0)	FASTBUS slot address.
GK	FASTBUS protocol.
INTRLEAVE_ACTIVE	Asserted by FSI at address time when the module is logically addressed and is enabled for address interleaving.
IRESET	Asserted by FSI during CSR#0 load if AD30 is set or during periods when the user asserts POWER_UP.
LA_MODE_EN	User signal indicating that advanced features of the FASTBUS protocol defined in the Los Alamos understanding are enabled.
LOAD_CSR0	Asserted by FSI for load command of CSR#0.
LOGICAL_ADDR_EN	Asserted by FSI when logical addressing is enabled with CSR#0<1>.
LOGICAL_WIDTH(4:0)	User signals defining the width of the internal address.
MS(2:0)	FASTBUS protocol.
NOT_VALID	Asserted by user to indicate the NTA address is not valid or the data received has a parity error.
NTA(31:0)	Next transfer address from FSI.
NTA_8TO29_NE_0	Asserted by the FSI when the NTA bits in the range 8 to 29 inclusive are not all reset.
NTA_ACCESS	Asserted by the FSI during secondary address cycles.
NTA_CARRY_IN	Asserted by user to indicate whether the NTA will advance when incremented.
NULL_CYCLE	Asserted by FSI following address connect-disconnect without a data cycle.
OE_BAD	User signal asserted when the user application permits the FSI to drive the data bus, BAD.
OE_NTA	User signal asserted when the user application permits the FSI to drive the next transfer address bus, NTA.
PA	FASTBUS protocol.
PE	FASTBUS protocol.
PARITY_ERROR	Asserted by FSI when the AD bus has wrong parity during write data cycles. User may use this signal to define NOT_VALID and REJECT. Latched for duration of data cycle.
POWER_UP	Asserted by user when power has been applied to the user module.
RD	FASTBUS protocol.
READ	Asserted by FSI to indicate read data cycle. The user application may use to set the direction for an AD bus buffer wherever direct connection between segment and FSI is not possible.
READ_CSR0	Asserted by FSI to indicate read command for CSR#0.
REJECT	Asserted by user to indicate that bad data or bad addresses will result in rejected data.
RESET_BUS	User signal asserted when RB and BH are both set.
RESPONSE_CTL_IN	Signal from user indicating that FSI may safely generate either AK(u) or DK(t).
RESPONSE_CTL_OUT	Signal to user indicating that subdivision needs to generate either AK(u) or DK(t). User logic may

	simply delay this signal and return the delayed version as RESPONSE_CTL_IN, or may implement more complex logic.
RUNNING	Asserted by FSI when module is enabled to run with CSR#0<2> = 1.
SET_ERROR_FLAG	Asserted by user when error to be reported through CSR#0<0>.
SET_HALT	Asserted by user when run mode latched by CSR#0<2> is to be cleared.
SET_SR	Asserted by user when SR is desired.
SET_SS3	User signal asserted when the user application wishes to generate SS = 3, the user defined completion code.
SR	FASTBUS protocol.
SR_FLAG	Asserted by FSI following receipt of SET_SR from user. Reset when CSR#0 loaded and bit 21 is set. Independent of the state of CSR#0<4>.
SS(2:0)	FASTBUS protocol.
TRANSFER_DONE	DK-like handshake from user. Generated for all data cycles, even those internal to the FSI chip. Prerequisite for RESPONSE_CTL_OUT.
TRANSFER_ENABLE	Asserted by FSI to indicate that the user must either accept or assert data.
TRANSFER_REQUEST	DS-like handshake to user. Request may be read, write, or cleanup cycle in either CSR or data space. User logic may simply delay this signal and return the delayed version as TRANSFER_DONE, or may implement more complex logic.
WT	FASTBUS protocol.
WT_EN	User signal which if set high will generate WT when BUSY asserted until TRANSFER_DONE indicates completion of transfer.

APPENDIX D

FASTBUS Slave Interface Schematics

AMCC BiCMOS Array Technology

1/16/92

BIDI 69, I 48, O 57

ALL ECL-10K

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.

3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER:

D. MACHEN, PE

DESIGN DATE:

1/16/92

PAGE:

1/1

TITLE:

FASTBUS SLAVE INTERFACE

TOP LEVEL FSI BLOCK

FILE NO:

SS/JV-10000

REV:

1

DATE:

1/21/92

8

7

6

5

4

3

2

1

8	7	6	5	4	3	2	1
SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA		DESIGNER: D. WACHEN, PE	DESIGN DATE: 1/16/92	PAGE: 3/ 10	TITLE: FAST0 U9 SLAVE INTERFACE INPUT PADS		FILE NO: SSI/N-10003
						REV: B *	DATE: 2/16/92 6/24

DESIGNER:	DESIGN DATE:	PAGE:	TITLE:	FILE NO:	REV:	DATE:
D. WAGEN, PE	1/21/92	5	FASTBUS SLAVE INTERFACE	SSI/N/10005	A	1/21/92
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA						
7	6	5	4	3	2	1

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA		DESIGNER: D. MACHEN, PE	DESIGN DATE: 1/21/92	PAGE: 6 / 10	TITLE: FASTBUS SLAVE INTERFACE BDI PADS 1/2 AD BUS	FILE NO: SS/JV-10006	REV: A	DATE: 1/21/92
8	7	6	5	4	3	2	1

DESIGNER:	DESIGN DATE:	PAGE:	TITLE:	FILE NO:	REV:	DATE:
D. MACHEN, PE	1/21/92	7/10	FASTBUS SLAVE INTERFACE	SSI/JV-10087	A	1/21/92
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA			B01 PAUS 1/2 AD BUS

DESIGNER: D. MACHEN, PE	DESIGN DATE: 1/21/92	PAGE: 9/18	TITLE: FASTBUS SLAVE INTERFACE	FILE NO: SSI/N-10000	REV: A	DATE: 1/21/92
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA			BID PADS 1/2 BAD bus			
6	5	4	3	2	1

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.

3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER:

D. WACHEN, PE

DESIGN DATE:

1/21/92

PAGE:

18/18

TITLE: FASTBUS SLAVE INTERFACE

BUFF INVERTERS

FILE NO:

SSI/JV-10018

REV:

A

DATE:

1/21/92

8

7

6

5

4

3

2

1

THESE MACROS ARE TO BE
PLACED ADJACENT TO THE
UE-59a ON AMCC_FSI/AMCC_FSI/8.DRW

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER: RON
D. MACHEN, PE

DESIGN DATE:
3/7/89

PAGE:
1/1

TITLE: FASTBUS SLAVE INTERFACE
USER SIDE MULTIPLEXER

FILE NO:
SSI/JV-10011

REV:
H

DATE:
8/8/92

8

7

6

5

4

3

2

1

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 1/24/92	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE "OR-AND COMBINER"	FILE NO: 55/AN-1006N	REV: B	DATE: 4/22/92
--	----------------------------	-------------------------	--------------	---	-------------------------	-----------	------------------

may need to expand enable logic depending on the final timing of LOAD_PB and DAT_CYC_SHT. see my design buffer_16/1.drw for details.

Q00 PARITY GENERATOR_EPA is low true in order to drive the pad. On an SS_ERROR during a read operation, parity is forced to one.

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.	DESIGNER: RON DUMACHEN, PE	DESIGN DATE: 10/5/88	PAGE: 1/2	TITLE: FASTBUS SLAVE INTERFACE	FILE NO: SS/W-10021	REV: K	DATE: 4/22/02
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA				PROTECTIVE BUFFER/PARITY GEN & CHECK

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.
3481 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER: RON
D. MACHEN, PE

DESIGN DATE:
18/6/88

PAGE:
2/2

TITLE: FASTBUS SLAVE INTERFACE
PARITY CHECK AND MISC CONTROL

FILE NO:
SSI/JV-18822

REV:
K

DATE:
4/22/92

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.

3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER: RON

D. MACHEN, PE

DESIGN DATE:

4/15/89

PAGE:

1/4

TITLE: FASTBUS SLAVE INTERFACE

BROADCAST ADDRESS STATE CONTROLLER

FILE NO:

SSU/N-10031

REV:

G

DATE:

4/28/92

DESIGNER:	DESIGN DATE:	PAGE:	TITLE:	FILE NO:	REV:	DATE:
D. MADSEN, PE	4/16/92	1/1	FASTBUS SLAVE INTERFACE	SSI/JN-18646	A	4/16/92
SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.						
3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.
3401 B TRINITY DR., LOS ALAMOS, NM 87544. USA

DESIGNER:
D. WACHEN, PE

DESIGN DATE:
3/24/89

PAGE:
1/2

TITLE: FASTBUS SLAVE INTERFACE
CSR0 & LOGIC

FILE NO:
SSI/JV-10051

REV:
M

DATE:
4/30/92

SMALL DELAY TO PREVENT DEV_SELECT S/R
FROM COLLIDING. 3/26/82

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: RON D. MACHEN, PE	DESIGN DATE: 3/26/80	PAGE: 2/4	TITLE: FASTBUS SLAVE INTERFACE STATE CONTROL LOGIC	FILE NO: SSI/JV-10062	REV: D	DATE: 4/3/82
--	--------------------------------	-------------------------	--------------	---	--------------------------	-----------	-----------------

8 7 6 5 4 3 2 1

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: RCM D. MACHEN, PE	DESIGN DATE: 3/28/80	PAGE: 4/4	TITLE: FASTBUS SLAVE INTERFACE AMCC ATTACH-TIME MODE AND NULL CYCLE	FILE NO: SS/JN-18864	REV: -R1-	DATE: 4/16/82
8	7	6	5	4	3	2	1

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: RON D. MACHEN, PE	DESIGN DATE: 4/12/89	PAGE: 1/4	TITLE: FASTBUS SLAVE INTERFACE DATA CYCLE CONTROL	FILE NO: SS/JV-10878	REV: F	DATE: 4/28/92
--	--------------------------------	-------------------------	--------------	--	-------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.	DESIGNER: RON	DESIGN DATE: 4/2/89	PAGE: 2/4	TITLE: FASTBUS SLAVE INTERFACE	FILE NO: SS/JN-10071	REV: F	DATE: 4/17/92
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	D. MADSEN, PE			DATA CYCLE CONTROL

DESIGNER: RON	DESIGN DATE: 11/16/00	PAGE: 3/4	TITLE: FASTBUS SLAVE INTERFACE	FILE NO: SSI/AN-19872	REV: F	DATE: 4/27/92
3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA			AMCC

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.

3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER: RON

D. WACHEN, PE

DESIGN DATE:

4/12/80

PAGE:

4/4

TITLE: FASTBUS SLAVE INTERFACE

AMCC

DATA CYCLE CONTROLS

FILE NO:

SSI/JV-10073

REV:

D

DATE:

4/7/92

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	ROW	DESIGN DATE: 6/7/89	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE AKK RESPONSE CONTROL	FILE NO: SSI/JN-10081	REV: H	DATE: 6/24/92
--	----------------------------	-----	------------------------	--------------	---	--------------------------	-----------	------------------

DESIGNER:	RON	DESIGN DATE:	4/12/88	PAGE:	2/2	TITLE:	FASTBUS SLAVE INTERFACE	FILE NO:	SS/JN-18022	REV:	P	DATE:	5/8/92
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	D. MADSEN, PE						SS AND WT CONTROL LOGIC

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.	DESIGNER:	DESIGN DATE:	PAGE:	TITLE:	FILE NO:	REV:	DATE:
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	D. MAGNEN, PE	1/24/82	1/2	FASTBUS SLAVE INTERFACE	SSI/N-18002N	A	1/24/82
				AM03 32 WIDE BUFFER WITH ENABLE HI TRUE			
8	7	6	5	4	3	2	1

36 LOAD DRIVE FOR Y OUTPUT.

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 1/28/79	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE LAMC3 4 TO 16 DECODER WITH ENABLE	FILE NO: SSI/N-1008BN	REV: B	DATE: 2/1/82
--	----------------------------	-------------------------	--------------	--	--------------------------	-----------	-----------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: RON D. MADSEN, PE	DESIGN DATE: 12/28/90	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE ANDC 8 BIT INCREMENTER	FILE NO: SS/JN-10000N	REV: B	DATE: 2/7/92
--	--------------------------------	--------------------------	--------------	--	--------------------------	-----------	-----------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: ROM D. MACHEN, PE	DESIGN DATE: 12/27/98	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE AMCC 32 BIT INDEPENDENT	FILE NO: SSI/N-10007N	REV: B	DATE: 2/7/92
--	--------------------------------	--------------------------	--------------	---	--------------------------	-----------	-----------------

DESIGNER:	D. MACHEN, PE	DESIGN DATE:	1/28/92	PAGE:	1/1	TITLE:	FASTBUS SLAVE INTERFACE	FILE NO:	SSJ/N-10000N	REV:	A	DATE:	1/28/92
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	6	7	8	9	10	11	12	13	14	15	16	17	18

E (CLOCK) HIGH = TRANSPARENT
E LOW - LATCH D TO Q

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 1/28/92	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE AMOC 8 WIDE LATCH WITH RESET LATR8	FILE NO: SSI/JV-10000N	REV: B	DATE: 2/7/92
--	----------------------------	-------------------------	--------------	---	---------------------------	-----------	-----------------

E (CLOCK) HIGH - TRANSPARENT
E LOW - LATCH D TO Q

DESIGNER: D. WACHEN, PE	DESIGN DATE: 1/28/82	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE	FILE NO: SS/JV-10818N	REV: B	DATE: 2/7/82
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA			AMCO 32 WIDE LATCH WITH RESET	LATCH32		
8	7	6	5	4	3	2

NOTE: HIGH TRUE RESET.

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.	DESIGNER:	D. MACHEN, PE	DESIGN DATE:	2/4/82	PAGE:	1/1	TITLE:	FASTBUS SLAVE INTERFACE	FILE NO:	SS/JN-10011N	REV:	D	DATE:	7/10/82
3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA							AMC32	32 WIDE D FLOP WITH RESET

NOTE: HIGH TRUE ENABLE.
S0, S1 AS IN ALS163

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.
3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER:
D. MACHEN, PE

DESIGN DATE:
12/26/98

PAGE:
1/1

TITLE: FASTBUS SLAVE INTERFACE
AMCC 32 WIDE 4 TO 1 MUX WITH ENABLE ME4T13

FILE NO:
SSV/JV-10012N

REV:
C

DATE:
6/8/02

-161-

NOTE: HIGH TRUE ENABLE

DESIGNER:	DESIGN DATE:	PAGE:	TITLE:	FILE NO:	REV:	DATE:
D. MACHEN, PE	2/5/92	1/1	FASTBUS SLAVE INTERFACE AMDC 32 WIDE, 2 TO 1 MUX	SSI/JN-10014N	A	2/6/92

1

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: RON . DFM	DESIGN DATE: 12/11/88	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE 8 WIDE INVERTER	FILE NO: SSI/JN-18016N	REV: B	DATE: 2/7/92
--	------------------------	--------------------------	--------------	--	---------------------------	-----------	-----------------

16INV
XA YN

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 2/7/92	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE AMCC 16 WIDE INVERTER	FILE NO: SSI/JV-10017N	REV: A	DATE: 2/7/92
--	----------------------------	------------------------	--------------	---	---------------------------	-----------	-----------------

8 7 6 5 4 3 2 1

2 CELLS
 UNSYMMETRIC 3 INPUT CONTROL
 DUE TO INVERSION PATH
 FASTER THAN THE MUX 163B, HOWEVER.

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 2/26/92	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE	FILE NO: SSI/AN-1881BN	REV: A	DATE: 2/26/92
--	----------------------------	-------------------------	--------------	-----------------------------------	---------------------------	-----------	------------------

1 2 3 4 5 6 7 8

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87644 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 2/6/92	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE AM02 16 TO 1 MUX	FILE NO: SSI/M-1001DN	REV: A	DATE: 4/24/92
--	----------------------------	------------------------	--------------	---	--------------------------	-----------	------------------

8 7 6 5 4 3 2 1

NOTE THE HIGH TRUE ENABLE HERE
A0, A1, AND E ARE 6 LOADS EACH

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. WAGNER, PE	DESIGN DATE: 4/24/92	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE DECODER 2 TO 3	FILE NO: SSI/N-10028N	REV: A	DATE: 4/24/92
--	----------------------------	-------------------------	--------------	--	--------------------------	-----------	------------------

DESIGNER:	DESIGN DATE:	PAGE:	TITLE:	FILE NO:	REV:	DATE:
D. MACHEN, PE	6/5/92	1/1	FASTBUS SLAVE INTERFACE AMCC 8 WIDE LATCH WITH RESET	SSI/JN-188821N	A	6/5/92

E (CLOCK) HIGH - TRANSPARENT
E LOG = LATCH D TO Q

DESIGNER: D. WACHEN, PE	DESIGN DATE: 6/5/92	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE AMCS 32 WIDE LATCH WITH RESET	FILE NO: SSI/JN-10022N	REV: A	DATE: 6/5/92
3401 B TRINITY DR., LOS ALAMOS, NM 87644 USA	6	5	4	3	2	1

NOTE: HIGH TRUE ENABLE.
S0, S1 LOW TRUE

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 12/28/90	PAGE: 1/1	TITLE: FASTBUS SLAVE INTERFACE AM03 32 WIDE 4 TO 1 MUX WITH ENABLE	FILE NO: SSI/IV-1824N	REV: D	DATE: 8/3/92
--	----------------------------	--------------------------	--------------	--	--------------------------	-----------	-----------------

NOTE: HIGH TRUE ENABLE FOR FSL
LO TRUE S0, S1

MM4T1B
X0
X1
X2
X3
YX
Z0
Z1
Z2
Z3

DESIGNER:	DESIGN DATE:	PAGE:	TITLE:	FILE NO:	REV:	DATE:
D. MACHEN, PE	12/28/98	1/1	FASTBUS SLAVE INTERFACE	MM4T1B	0	8/3/92
3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA			AMCC 8 WIDE 4 TO 1 MUX	SS/JV-10013N		

APPENDIX E

FASTBUS Slave Interface Schematics

Raytheon ECL Array Technology

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.

3401 B TRINITY DR., LOS ALAMOS, NM 87644 USA

DESIGNER: D. MACHEN, PE

DESIGN DATE: 10/77/88

PAGE: 1/1

TITLE: FASTBUS BULK STORAGE MODULE

TOP LEVEL FSI BLOCK

FILE NO: SSI/JV-10000

REV: M

DATE: 10/9/90

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 3/5/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE FS INPUT & BDIR PADS	FILE NO: SSI/JV-10000E1	REV: L DATE: 8/11/88
--	----------------------------	------------------------	--------------	--	----------------------------	-------------------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. MACHEN, PE	DESIGN DATE: 4/12/80	PAGE: 2/2	TITLE: FASTBUS BULK STORAGE MODULE FSI OUTPUT PADS & BIDIR	FILE NO: SS/JN-1000002	REV: A	DATE: 10/1/80
--	----------------------------	-------------------------	-----------	--	---------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: D. WAGEN, PE	DESIGN DATA: 10/6/88	PAGE: 2/2	TITLE: FASTBUS BULK STORAGE MODULE BUFFER-MUX BAD TRANSCEIVERS	FILE NO: SSI/VV-10012	REV: E	DATE: 7/7/89
--	---------------------------	-------------------------	--------------	--	--------------------------	-----------	-----------------

SCIENTIFIC SYSTEMS INTERNATIONAL	DESIGNER:	DESIGN DATA:	PAGE:	TITLE:	FILE NO:	REV:	DATE:
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	D.J. MACHEN, PE	6/16/88	2/4	FASTBUS BULK STORAGE MODULE RAYTHEON ARRAY - PRIM. ADDR	SSI/N-18003	C	7/6/88
8	6	5	4	3	2	1

DESIGNER: DRM	DESIGN DATE: 3/1/89	PAGE: 3/4	TITLE: SUB-ADDR COMPARATOR	FILE NO: SS/00-10004	REV: A	DATE: 3/26/91
3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.	DESIGNER: DRM	DESIGN DATE: 3/1/89	PAGE: 4/4	TITLE: FASTBUS BULK STORAGE MODULE	FILE NO: SS/JV-18868	REV: A	DATE: 3/26/91
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	R. NELSON, PHD			LOGICAL ADDRESS COMPARATOR

SCIENTIFIC SYSTEMS INTERNATIONAL
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER:
R. NELSON, PH.D.

DESIGN DATA:
3/7/89

PAGE:
1/1

TITLE: FASTBUS BULK STORAGE MODULE
USER APPLICATION BUFFER/MULTIPLEXER

FILE NO:
SS/JV-10011

REV:
E

DATE:
7/14/88

ORAND1

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3481 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: DRM R. NELSON, PH.D.	DESIGN DATE: 4MAR69	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE OR/AND GATE IN THE STYLE OF THE RAYTHEON NNR42	FILE NO: SSI/N-19814	REV: A	DATE: 3/26/81
--	-----------------------------------	------------------------	--------------	--	-------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: R. NELSON, PHD	DESIGN DATE: 3/1/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE 8 WIDE, 4 TO 1 MULTIPLEXER WITH ENABLE	FILE NO: SSI/4V-18812	REV: A	DATE: 3/26/91
--	-----------------------------	------------------------	--------------	--	--------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3491 B TRINITY DR., LOS ALAMOS, NM 87644 USA	DESIGNER: R. NELSON, PHD	DESIGN DATE: 3/1/89	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE 8 WIDE, 4 TO 1 MULTIPLEXER WITH ENABLE	FILE NO: SS/JV-18M13	REV: A	DATE: 3/26/91
--	-----------------------------	------------------------	--------------	--	-------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER:
R. NELSON, PH.D.

DESIGN DATA:
10/6/88

PAGE:
1/1

TITLE: FASTBUS BULK STORAGE MODULE
PROTECTIVE BUFFER/PARTY CHECKER/PARTY GEN BUFFER_FB

FILE NO:
SSI/JV-10021

REV:
C

DATE:
8/7/90

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: R. NELSON, P.H.D.	DESIGN DATE: 15APR89	PAGE: 1/3	TITLE: FASTBUS BULK STORAGE MODULE BROADCAST ADDRESS STATE MACHINE CONTROLLER	FILE NO: SS/JV-10031	REV: D	DATE: 2/12/01
--	--------------------------------	-------------------------	--------------	---	-------------------------	-----------	------------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: R. NELSON, PH.D.	DESIGN DATE: 16APR89	PAGE: 3/3	TITLE: FASTBUS BULK STORAGE MODULE CSR 7	FILE NO: SSI/JN-10033	REV: B	DATE: 4/17/89
8	7	6	5	4	3	2	1

3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: DRM R. NELSON, PHD	DESIGN DATE: 3/1/88	PAGE: 1/1	TITLE: FASTBUS BULK STORAGE MODULE 32 WIDE, 4 TO 1 MUX	FILE NO: SSI/AV-18844	REV: A	DATE: 3/28/81
--	---------------------------------	------------------------	--------------	---	--------------------------	-----------	------------------

DESIGNER:	DRM	DESIGN DATE:	3/1/68	PAGE:	1/1	TITLE:	FASTBUR BULK STORAGE MODULE 32 WIDE COMPARE	FILE NO.	881/N-10043	REV:	A	DATE:	3/28/61
3491 B TRINITY DR., LOS ALAMOS, NM 87544 USA	6	7	8	9	10	11	12	13	14	15	16	17	18

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.	DESIGNER: D. MACHEN, PE	DESIGN DATE: 3/24/89	PAGE: 1/2	TITLE: FASTBUS BULK STORAGE MODULE	FILE NO: SS/JN-10861	REV: J	DATE: 3/6/91
3401 B TRINITY DR., LOS ALAMOS, NM 87644 USA

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: R. NELSON, PH.D.	DESIGN DATE: 22MAY83	PAGE: 1/3	TITLE: FAST BUS BULK STORAGE MODULE FSI STATE MACHINE CONTROLLER	FILE NO: SSI/JV-10061	REV: E	DATE: 9/8/80
--	-------------------------------	-------------------------	--------------	---	--------------------------	-----------	-----------------

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA	DESIGNER: R.D. NELSON, PH.D.	DESIGN DATE: 26MAR89	PAGE: 2/3	TITLE: FASTBUS BULK STORAGE MODULE STATE CONTROL LOGIC	FILE NO: SS/JV-10062	REV: C	DATE: 4/21/91
--	---------------------------------	-------------------------	--------------	---	-------------------------	-----------	------------------

8	7	6	5	4	3	2	1		
SCIENTIFIC SYSTEMS INTERNATIONAL, LTD. 3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA			DESIGNER: R.D. NELSON, PH.D.	DESIGN DATE: 26MAR83	PAGE: 3/3	TITLE: FASTBUS BULK STORAGE MODULE FS STATE-CTL LATCH FOR ATTACH TIME MODE	FILE NO: SS/JV-10063	REV: E	DATE: 9/11/80

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER:
R. NELSON, PHD

DESIGN DATE:
4/12/89

PAGE:
3/3

TITLE: FASTBUS BULK STORAGE MODULE
DATA_CYCLE CONTROL

FILE NO:
SSI/JV-10072

REV:
B

DATE:
4/14/91

8

7

6

5

4

3

2

1

SCIENTIFIC SYSTEMS INTERNATIONAL, LTD.
3401 B TRINITY DR., LOS ALAMOS, NM 87544 USA

DESIGNER:
R. NELSON

DESIGN DATE:
11/27/98

PAGE:
2/2

TITLE: FASTBUS BULK STORAGE MODULE
SSWT RESPONSE LOGIC

FILE NO:
SSI/JV-10092

REV:
B

DATE:
8/28/98

8

7

6

5

4

3

2

1

APPENDIX F
FASTBUS Slave Interface Technical Paper
Technical Brochure

*conf paper
cycled separately
ds*

SSI

MICROFILM
THIS PAGE

Scientific Systems International

3491 B Trinity Drive
Los Alamos
New Mexico 87544 USA
(505) 662-6712

November 13, 1995

To: Department of Energy
Chicago Operations Office

From: Donald R. Machen

Ref: 80454 and 80599

RECEIVED
DEC 08 1995
OSTI

I hereby authorize the deletion of "Rights in SBIR Data" in the Final Technical Reports of the reference contracts.

Donald R. Machen