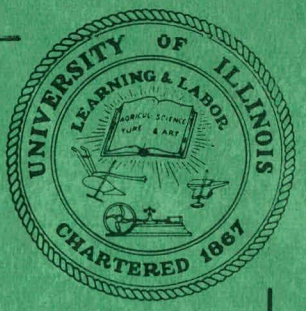

FASTBUS Segment Driver Microcode Description*

David Lesny
October, 1981
Updated June, 1983

This document reflects microcode version 5(126)

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

DEPARTMENT OF PHYSICS

LOOMIS LABORATORY OF PHYSICS

1110 W. GREEN STREET

URBANA, ILLINOIS 61801

TR-225

DISCLAIMER

DOE/ER/01195-477

DOE/ER/01195--477

DE83 017972

COO-1195-477

FASTBUS Segment Driver Microcode Description*

David Lesny
October, 1981
Updated June, 1983

University of Illinois at Urbana-Champaign
Department of Physics
High Energy Physics Group
Urbana, Illinois

This document reflects microcode version 5(126)

DISCLAIMER

* Work supported in part by the Department of Energy,
contract DE-AC02-76ER01195

Pacs Index: 06.50.Mk

Table of Contents

CHAPTER 1

FASTBUS SEGMENT DRIVER

The FASTBUS Segment Driver, here after referred to as the FSD, is a list-driven, microcoded interface between the UNIBUS of a PDP-11 system and the FASTBUS. The list structure used by the FSD allows the programmer on the PDP-11 to program a sequence of data transfers to take place without the aid or intervention of the PDP-11. This allows the FASTBUS to be driven at FSD rates, independent of the PDP-11 processor.

Due to the difference in speed between the FASTBUS and UNIBUS, the major goal of the FSD was to provide an interface which could transfer data on FASTBUS without significantly reducing the bandwidth in a multi-master system. This was accomplished by bursting data on the FASTBUS through a 256 word fast buffer internal to the FSD. Data can be transferred at near FASTBUS rates through this memory and only moved on the UNIBUS when the FSD is not master of FASTBUS. This allows other masters in the same system to transfer their data while the FSD is moving data on the slower UNIBUS.

1.1 FSD FEATURES

The FSD has the ability to:

1. Perform all FASTBUS cycles except pipelined transfers, in either complete operations (arbitration, primary address cycle, secondary address cycle, data cycles, termination) or individual primitive cycles.

2. Provide complete status on the execution of all FASTBUS Operations and primitive cycles.

3. Be programed for various responses to errors on FASTBUS. These responses include terminating the list, terminating the element, ignoring the error or automatically retrying a programable number of times.

4. Accept a single UNIBUS buffer through which I/O on the FASTBUS takes place.

5. Perform the same FASTBUS Operation on a group of FASTBUS devices with a single opcode.

6. Transfer data between two FASTBUS devices without moving the data onto the UNIBUS. This allows FASTBUS transfers whose rates are dictated only by the master/slave response times.

7. Automatically break block transfers into bursts of between 1 and 256 words to help prevent locking out other masters in the system.

1.2 PROGRAMMING THE FSD

The FSD is a list driven interface. The FSD takes instructions from a list and returns status to a status block. Data is tranferred to and from a UNIBUS buffer. There are five parts involved in controlling the FSD.

1. UNIBUS I/O-Page Registers

   The I/O-Page registers are used to start and stop the FSD and to report the state of the FSD.

2. Control Block

   The control block contains parameters for the entire list as well as the addresses of the list, status block and UNIBUS buffer.

3. List Block

   The list block contains a sequence of instructions which the FSD is to follow. The FSD executes these instructions, or list elements, until either the end of the list is reached or a fatal error causes an abort.

4. Status Block

   As the FSD executes the list, information about the execution of the individual elements is returned to the status block.

5. UNIBUS Buffer

   The UNIBUS buffer, which is defined only once per list, is a block of memory where the FSD reads and writes the data being transferred. The location of the UNIBUS buffer and its length are defined in the control block.

Programming the FSD involves setting up a control block and list, allocating space for the status block and UNIBUS buffer, and filling in the UNIBUS buffer with data to write to the FASTBUS. The PDP-11 tells the FSD where the control block is within UNIBUS memory through the I/O-Page registers. The FSD reads the control block into internal memory. Once the control block is read, the FSD begins executing the list elements contained within the list. Status information about the execution is returned to the status block. When the FSD completes the execution of the list, a PDP-11 interrupt can be generated by the FSD.

1.3 UNIBUS VS FASTBUS BUFFERS

Most data transfers on FASTBUS move data through the UNIBUS memory. Because of this, the FSD is geared mostly toward UNIBUS to FASTBUS operations. For example the UNIBUS buffer is defined in the control block and remains defined for the entire list. There are also several FSD instructions (opcodes) which allow the user to manipulate the UNIBUS buffer pointer. However in some cases, the user wants to move data between two FASTBUS devices. This can be accomplished by moving the data from the first FASTBUS device into the UNIBUS buffer, reseting the UNIBUS buffer pointer, and finally writing the data to the second FASTBUS device. Since the UNIBUS is relatively slow in comparision to the FASTBUS, and because the data is being moved twice (from FASTBUS to UNIBUS, then UNIBUS to FASTBUS), the time to execute this type of operation would be very large. To help speed up these types of operations, the FSD supports a second data transfer mechanism called FASTBUS Transfer Addressing, sometimes referred to as FASTBUS to FASTBUS transfers.

When a FASTBUS to FASTBUS transfer is to be performed, the user first defines the FASTBUS Transfer Device with an FSD opcode. This FASTBUS Transfer Device can then be used, at the discretion of the user, in place of the UNIBUS buffer. The data is never moved into the UNIBUS memory. All data motion takes place on FASTBUS between the two FASTBUS devices and the FSD. This allows the data to be transferred at FSD rates. These types of data transfers are fully explained in chapter 9.

1.4 MULTIPLE DEVICE ADDRESSING

Most operations performed on FASTBUS by the FSD require only one FASTBUS primary address. At times it is useful to perform the same operation on a group of devices. The FSD provides a mechanism for doing this, called Multiple Device Addressing. This feature allows the user to define a group of FASTBUS primary address modules and then perform an operation, via one request, on this group of addresses. See chapter 8 for a complete description of Multiple Device Addressing.

1.5 DIFFERENCES IN ADDRESSING AND WORD SIZES

The UNIBUS has a 16-bit word and 16, 18 or 22-bit addressing (dependent on the CPU model), whereas FASTBUS has a 32-bit word and 32-bit addressing. Also the UNIBUS uses 8-bit byte addressing, whereas FASTBUS uses word addressing. There is therefore a 4 to 1 address map formed between the UNIBUS and FASTBUS. These differences in word size as well as addressing can lead to the possibility of confusion when talking about addresses, offsets, word counts, etc. In general the following statements describe how the microcode deals with these differences.

1. If the word(s) are in the UNIBUS memory, they are 16-bit words.

2. If the word(s) are in a FASTBUS device, they are 32-bit words.

3. Addressing on the UNIBUS is in terms of bytes, however the least significant bit of the address is always ignored (addressing is always on a word boundary). Likewise, offsets into UNIBUS memory are always given in bytes.

4. Addressing on the FASTBUS is in terms of 32-bit words. Likewise, offsets in FASTBUS address space are given in terms of 32-bit words.

5. Word counts are always in terms of the number of 16-bit words. If 32-bit words are being transferred, the word count specified must be double the number of words to be transferred. Likewise the count returned in the status element is the number of 16-bit words transferred.

6. Offsets in list elements, and those returned in the status elements are in terms of 16-bit words.

7. A 32-bit word in UNIBUS memory occupies two 16-bit words in sequential adddresses. The least significant 16-bits (LSB, bits 0 through 15) occupy the first word, and the most significant 16-bits (MSB, bits 16 through 31) occupy the second word.

8. A 16-bit word in UNIBUS memory occupies only one word. This word will map into the least significant 16 bits of a 32-bit word on FASTBUS. When reading data from FASTBUS and storing in the UNIBUS buffer, the most significant bits are never stored. When writing data on FASTBUS which has been take from the UNIBUS buffer, the most significant bits are always zeroed. This allows for packed data when doing 16-bit mode transfers on FASTBUS.

CHAPTER 2

UNIBUS I/O-PAGE REGISTERS

There are thirteen UNIBUS I/O-Page registers associated with the FSD. The PDP-11 can read and write all thirteen of these registers, however only eleven are used by version 5 of the microcode. Of these eleven, only three ever need be accessed by the PDP-11. These I/O-Page registers are used for control and status reporting of the FSD. The other eight are used for debugging new versions of the microcode and are not needed for controling the FSD. The offsets and description of each I/O-Page register are given in table 2-1.

Prior to executing a list, the user must inform the FSD where the list resides in memory. This is done by supplying the 18-bit address of the control block in registers FSDCTL and FSDCSR. Upon completion of the list, status is returned by the FSD in FSDCSR. Register FSDHCS is used to control the FSD.

The offset is from the base address strapped on the UPI interface board. LSB stands for the least significant 16 bits (bits 0 through 15). MSB stands for the most significant 16 bits (bits 16 through 31).

Offset	Name	Description
0	FSDCTL	LSB of the control block address
2	FSDCSR	Software control/status register
4	FSDMSC	Microcode status code
6		unused FSD register
10	FSDPRM	List parameters
12		unused FSD register
14	FSDBAL	LSB of the buffer address
16	FSDBAH	MSB of the buffer address
20	FSDLAL	LSB of the list block address
22	FSDLAH	MSB of the list block address
24	FSDSAL	LSB of the status block address
26	FSDSAH	MSB of the status block address
30		UPI register
32		UPI register
34	FSDHCS	Hardware control/status register

Table 2-1. I/O-Page locations and descriptions

2.1 FSDHCS - HARDWARE CONTROL/STATUS REGISTER.

This register contains bits which control the FSD and return status on the current state of the FSD.

Bit	Name	Description
15		unused
14		unused
13		unused
12		unused
11		unused
10		unused
9		unused
8		unused
7	HCSRDY	FSD ready flag
6	HCSIE	Interrupt enable
5		unused
4		unused
3		unused
2		unused
1	HCSHLT	FSD halt control
0	HCSXCT	Execute a new list

2.1.1 HCSRDY (RO) - FSD Ready

When this bit is set, the FSD is in an idle state and is polling the execute bit (HCSXCT). When the PDP-11 sets HCSXCT, the FSD will begin executing the list pointed to by FSDCTL and FSDCSR and HCSRDY will be cleared. When the FSD finishes execution of the list, HCSRDY will again be set. If execution is halted by the setting of HCSHLT in FSDHCS, the FSD will again set HCSRDY when it becomes idle. This is a read only bit to the PDP-11.

2.1.2 HCSIE (RW) - Interrupt Enable

If this bit is set, the FSD will generate a vectored interrupt on the PDP-11 whenever the ready bit (HCSRDY) becomes set. If ready is set when the PDP-11 sets this bit, an interrupt will also be generated.

2.1.3 HCSHLT (RW) - Halt The FSD

The setting of this bit will cause an active FSD to suspend the execution of a list. The FSD will stop executing the list when the current list element is completed, even if it is FASTBUS master. When the FSD halts, it outputs the status block header, sets the ready bit (HCSRDY), and polls for further instructions from the PDP-11. Should the PDP-11 clear the halt bit, the FSD will resume execution of the list. However if the execute bit (HCSXCT) is set, the FSD will begin execution of a new list by reading a new control block. Setting HCSXCT will automatically clear HCSHLT.

2.1.4 HCSXCT (WO) - Execute A New List

When set by the PDP-11, the FSD will begin executing a list as defined by the control block pointed to by FSDCTL and FSDCSR. If the FSD is not idle when this bit is set, the execute command will be ignored. This is a write only bit to the PDP-11 and is always read as a zero.

2.2 FSDCTL - CONTROL BLOCK ADDRESS

This register contains the least significant 16 bits of the control block address. Address bits 16 and 17 are contained in FSDCSR. When the HCSXCT bit in FSDHCS is set, the FSD uses the contents of this register and FSDCSR to build a pointer to the control block.

2.3 FSDMSC - MICROCODE STATUS CODE

This register is used primarily for debugging the microcode. The microcode returns a number in this register indicating its current state. The current state codes are

Code	Name	Description
0	MSCIDL	Idle
1	MSCCTL	Loading the control block
2	MSCXLE	Setting up to execute a list element
3	MSCDON	Terminating a list
4	MSCWES	Terminating a list element
5		unused
6	MSCFMC	Waiting for FASTBUS master cycle complete
7	MSCFAC	Waiting for FASTBUS address cycle complete
10	MSCFDC	Waiting for FASTBUS data cycle complete
11	MSCIHB	Idle but holding FASTBUS mastership

2.4 FSDPRM - LIST PARAMETERS

This register contains the contents of the control block entry CTLPRM while the FSD is executing a list. This register is for maintenance only.

2.5 FSDBAL/FSDBAH - BUFFER ADDRESS

These registers contain the 18-bit buffer address which the FSD is using for the currently executing list. FSDBAL contains the least significant 16 bits. FSDBAH contains the extended address bits 16 and 17. This register is for maintenance only.

2.6 FSDLAL/FSDLAH - LIST BLOCK ADDRESS

These registers contain the 18-bit list block address which the FSD is currently executing. FSDLAL contains the least significant 16 bits. FSDLAH contains the extended address bits 16 and 17. This register is for maintenance only.

2.7 FSDSAL/FSDSAH - STATUS BLOCK ADDRESS

These registers contain the 18-bit status block address where the FSD reports status on the currently executing list. FSDSAL contains the least significant 16 bits. FSDSAH contains the extended address bits 16 and 17. This register is for maintenance only.

2.8 FSDCSR - SOFTWARE CONTROL/STATUS REGISTER.

FSDCSR contains information for controlling the execution of the FSD as well as the status of the FSD upon termination of a list. All bits within this register can be read and written by the PDP-11, however only those bits which are used to control the FSD should be changed by the PDP-11. The bits which return status are only valid when the FSD is in the ready state.

Bit	Name	Description
15	CSRERR	Stopped by a fatal error
14	CSRHLT	Stopped by a HALT command (in FSDHCS)
13	CSRLEI	Stopped by a list element opcode 030
12	CSRLRE	Stopped by the limit register being exceeded
11	CSRSBO	Stopped by the status block overflowing
10	CSRHFB	Holding FASTBUS mastership
9	CSRHFA	Holding FASTBUS address lock
8		unused
7	CSRWRN	A non-fatal error has occured
6	CSRNXC	UNIBUS NXM during a control block read
5	CSRNXS	UNIBUS NXM during a status block write
4	CSRNXL	UNIBUS NXM during a list element read
3	CSRNXB	UNIBUS NXM during a buffer access
2	CSRRST	Software reset
1	CSRA17	Bit 17 of the control block address
0	CSRA16	Bit 16 of the control block address

2.8.1 CSRERR (RO) - Stopped By A Fatal Error

The FSD aborted execution of the list because it encountered a fatal error. The type of error is indicated in either FSDCSR or the status block header word. The FSD cannot continue and must be restarted by loading a new list.

2.8.2 CSRHLT (RO) - Stopped By A HALT Command

The FSD has suspended execution of the current list because the PDP-11 set bit HCSHLT in the FSDHCS register. When the halt bit is cleared the FSD will continue where it stopped in the current list. If the execute bit is set (HCSXCT), the FSD will abort the current list and begin the execution of a new list as pointed to by FSDCTL and FSDCSR.

2.8.3 CSRLEI (RO) - Stopped By A List Element Opcode 030.

The FSD has suspended execution of the current list because an opcode 030, list element interrupt, was executed. The FSD is in the same state as if the HCSHLT bit was set in FSDHCS. When the PDP-11 clears bit HCSHLT, the FSD will continue execution of the current list. Should the PDP-11 set bit HCSXCT instead, the FSD will begin the execution of a new list as pointed to by FSDCTL and FSDCSR. No further action is taken on the old list.

2.8.4 CSRLRE (RO) - Stopped By The Limit Register Being Exceeded

The limit register (as given by CTLBLM of the control block, section 3.4) was exceeded by the execution of the last list element. The list pointer offset (STLLPL/STLLPH) contained in the status header block points to the list element immediately following the element which moved the buffer pointer past the limit.

2.8.5 CSRSBO (RO) - Stopped By The Status Block Overflowing

Not enough room was allocated in the status block body for the list to be executed. The list element which caused this error is NOT executed. The list pointer offset (STLLPL/STLLPH) contained in the status header block points to the list element which would have written its status beyond the status block had it been executed.

2.8.6 CSRHFB (RO) - Holding FASTBUS Mastership

The FSD is the current master of FASTBUS even though it is in the idle state. This condition will result if the last list element executed had the option bits OPCHFB or OPCHFA set and the FSD was FASTBUS master. It will also occur if the list aborted with a fatal error while the FSD was bus master and the parameter bit PRMHFE was set in the parameter word CTLPRM. To release mastership, either the software reset bit, CSRRST, must be set or a new list must be executed by the FSD.

2.8.7 CSRHFA (RO) - Holding FASTBUS Address Lock

The FSD is connected to a FASTBUS device. The address of the device is returned in the status header block (STLPAL/STLPAH). This condition will result if the last element executed had the option bit OPCHFA set and the FSD was connected to a device. It will also occur if the list aborted with a fatal error while a device was connected and the option bit PRMHFE was set in the parameter word CTLPRM. A software reset will force the FSD to release the device.

2.8.8 CSRWRN (RO) - A Non-fatal Error Has Occurred

Some list element had a non-fatal error. The PDP-11 must search the status block to determine which element(s) had the warning. A warning is either a non-fatal parity error or a non-fatal slave status code.

2.8.9 CSRNXC (RO) - UNIBUS NXM During The Control Block Read

The control block address given in FSDCTL and FSDCSR does not exist in UNIBUS memory. The FSD immediately becomes ready. The FSD retains control of the FASTBUS if it was currently FASTBUS master.

2.8.10 CSRNXS (RO) - UNIBUS NXM During A Status Block Write

The FSD attempted to output the status for either the list or a list element to a non-existent location in UNIBUS memory. Execution of the list is aborted and no further status is output. The FSD remains FASTBUS master if it was bus master and bit PRMHFE in word CTLPRM was set.

2.8.11 CSRNXL (RO) - UNIBUS NXM During A List Element Read

The FSD attempted to read a list element from a non-existent memory location. The list element offset pointer in the status block header points at the element where the FSD had the problem. The list is aborted and the status block header is output. The FSD remains FASTBUS master if it was bus master and bit PRMHFE in word CTLPRM was set.

2.8.12 CSRNXB (RO) - UNIBUS NXM During A Buffer Access

The FSD got a non-existent memory error while trying to read or write the buffer on the UNIBUS. The list is aborted and the status block header is output. The FSD remains FASTBUS master if it was bus master and bit PRMHFE in word CTLPRM was set.

2.8.13 CSRRST (RW) - Software Reset

If this bit is set when the FSD begins execution, a software reset is performed instead of a list execution. The control block address in FSDCTL is ignored. If the FSD is FASTBUS master, a release of the bus is done. The status bits in FSDCSR are reset. No other status information is returned. If this bit was set while the FSD was halted (setting HCSHLT or opcode 030), the current list is aborted. The status block contains valid information up to the point of termination since the halting of the FSD forces all status to be output to the block.

2.8.14 CSRA17/CSRA16 (RW) - Control Block Extended Address Bits

CSRA17 and CSRA16 are the extended address bits of the control block address. These bits, along with the contents of FSDCTL, form an 18 bit address which points to the first word of the control block.

CHAPTER 3

CONTROL BLOCK

The control block is a seven 16-bit word block of memory constructed by the programmer. It contains parameters needed by the FSD to define the list to be executed, the UNIBUS buffer and the location in memory where status on the execution of the list is to be stored. Table 3-1 defines the offsets into the control block where each parameter is stored. The 18-bit address of the first word of the control block is given to the FSD in the I/O-Page registers FSDCTL and FSDCSR. When bit HCSXCT is set in the FSDHCS register, the FSD reads the control block into internal RAM.

Offset	Name	Description
0	CTLPRM	List parameters
2	CTLBFA	LSB of the UNIBUS buffer address
4	CTLBFL	UNIBUS buffer length
6	CTLBLM	UNIBUS buffer limit
10	CTLLBA	LSB of the list block address
12	CTLSBA	LSB of the status block address
14	CTLSBL	Status block length

Table 3-1. Control block format

3.1 CTLPRM - LIST PARAMETERS

Options which are on a list wide basis are contained in CTLPRM. Extended address bits for the buffer, list and status block addresses are also contained in this register.

Bit	Name	Description
15	PRMNSE	No status elements are to be written
14		unused
13		unused
12		unused
11		unused
10		unused
9		unused
8		unused
7	PRMB17	Bit 17 of the UNIBUS buffer address
6	PRMB16	Bit 16 of the UNIBUS buffer address
5	PRML17	Bit 17 of the list address
4	PRML16	Bit 16 of the list address
3	PRMS17	Bit 17 of the status block address
2	PRMS16	Bit 16 of the status block address
1	PRMHFE	Hold FASTBUS mastership after a fatal error
0	PRMWEN	Write enable the UNIBUS buffer

3.1.1 PRMNSE - No Status Elements Are To Be Written

If PRMNSE is set, the FSD will suppress the writing of status elements for each list element executed. The status header will still be written upon the termination of the list.

3.1.2 PRMB17/PRMB16 - UNIBUS Buffer Extended Address Bits

PRMB17 and PRMB16 are the extended address bits for the UNIBUS buffer address. These bits, along with the contents of CTLBFA, form an 18-bit address which points to the first word of the buffer.

3.1.3 PRML17/PRML16 - List Extended Address Bits

PRML17 and PRML16 are the extended address bits for the list address. These bits, along with the contents of CTLLBA, form an 18 bit address which points to the first word of the list.

3.1.4 PRMS17/PRMS16 - Status Block Extended Address Bits

PRMS17 and PRMS16 are the extended address bits for the status block address. These bits, along with the contents of CTLSBA, form an 18 bit address which points to the first word of the status block for the list.

3.1.5 PRMHFE - Hold FASTBUS Mastership After An Error

The setting of this bit will cause the FSD to maintain FASTBUS mastership and the current device connection (if they are present) when the list terminates due to a fatal error. If this bit is clear the FSD will release any connected device and bus mastership prior to becoming ready. Bits CSRHFB and CSRHFA in FSDCSR reflect the current state the FSD with respect to the FASTBUS.

3.1.6 PRMWEN - Write Enable The UNIBUS Buffer

The UNIBUS buffer is always readable by the FSD. However this bit must be set for the FSD to be able to write into the buffer. Should this bit not be set and the FSD attempt to execute a list element which requires writing the buffer, the list will be terminated with a write protect error.

3.2 CTLBFA - LSB OF THE UNIBUS BUFFER ADDRESS

CTLBFA holds the least significant 16 bits of the UNIBUS buffer address. This entry along with bits PRMB17/PRMB16 of CTLPRM form the 18-bit address of the UNIBUS buffer to be used by the list. This address is the initial internal buffer address used by the FSD. Whenever data is transferred through the FSD to or from the UNIBUS buffer, the internal pointer is incremented by the number of words transferred. Special function list elements are provided which move the buffer pointer relative to the current location or relative to the initial address. These elements will not allow the pointer to be moved beyond the buffer address. If the pointer is moved beyond the limit register address, and attempt to execute another opcode will result in a limit register exceeded error.

3.3 CTLBFL - UNIBUS BUFFER LENGTH

CTLBFL holds the length of the UNIBUS buffer being used. The length is given in terms of 16-bit words. The FSD will not attempt to access any locations beyond the buffer length. Should a list element attempt to go beyond this upper bound, the FSD will halt the list with a buffer overflow error.

3.4 CTLBLM - UNIBUS BUFFER LIMIT

CTLBLM holds the length which sets a pointer limit in 16-bit words. Should the internal buffer address pointer be moved beyond the limit address, the FSD will not begin the next element in the list. The list is aborted with a limit register exceeded error (CSRLRE is set in FSDCSR).

3.5 CTLLBA - LIST BLOCK ADDRESS

CTLLBA holds the least significant 16 bits of the list address. The most significant 16 bits are taken from the parameter register (CTLPRM), bits PRML16 and PRML17. The list is decribed in chapter 4.

3.6 CTLSBA - STATUS BLOCK ADDRESS

CTLSBA holds the least significant 16 bits of the status block address. The most significant bits are taken from the parameter register (CTLPRM), bits PRMS16 and PRMS17. The status block is described in chapter 5.

CHAPTER 4

LIST FORMAT

A "list" is a group of contiguous, multi-word, fixed length "list elements" which resides in UNIBUS memory. Each list element is eight 16-bit words in length. The list element consists of an opcode which defines the task the FSD is to perform, option bits which modify the task, and parameters needed by the FSD to perform the task. Some opcodes do not use every word within the element, however the space must still be allocated within the list.

The list is read only to the FSD. The status of the execution of the list is returned in the status block. The FSD executes an element one at a time until either a termination element is found, or a fatal error occurs. The format of a list element is given in table 4-1.

Offset	Name	Description
0	LSTOPC	Opcode and option bits
2	LSTOPT	Option bits
4	LSTPAL	LSB of the FASTBUS primary address
6	LSTPAH	MSB of the FASTBUS primary address
10	LSTSAL	LSB of the FASTBUS secondary address
12	LSTSAH	MSB of the FASTBUS secondary address
14	LSTWCL	LSB of the word count or immediate data
16	LSTWCH	MSB of the word count or immediate data

Table 4-1. List element format

4.1 LSTOPC - OPCODE AND OPTION BITS

This word defines the task which the FSD is to perform. The option bits provide a means of modifing the meaning of the opcode.

Bit	Name	Description
15	OPCPER	Parity error reponse code
14	OPCPER	Parity error response code
13	OPCPER	Parity error response code
12	OPCPER	Parity error response code
11		unused
10	OPCFTS	FASTBUS transfer secondary addressing
9	OPCWID	Write immediate data
8	OPCHLF	Half word transfer
7	OPCSTD	Standard vs Special function opcode
6		opcode
5		opcode
4		opcode
3		opcode
2		opcode
1		opcode
0		opcode

4.1.1 OPCPER - Parity Error Response Code

The parity error reponse code is a 4 bit encoded field used by the FSD to determine the type of action to be taken when parity errors are detected. Unlike the 32-bit error response words in the FSD's system parameter block, this reponse code is on an element by element basis. It controls parity errors during secondary address cycles as well as data cycles. Each of the 16 possible response codes are identical in function to the 16 response codes in the error response word.

4.1.2 OPCFTS - FASTBUS Transfer Secondary Addressing

This option bit is used by the special function opcode (060) which loads the FASTBUS Transfer Device address. It indicates that a secondary address cycle must be performed after each primary address cycle to FASTBUS Transfer Device. For the other FASTBUS devices, the secondary address bit is actually part of the opcode. This option bit is ignored by all other opcodes.

4.1.3 OPCWID - Write Immediate Data

This option bit is used only by those opcodes which do single word writes to the FASTBUS. When set, the data word to be written to FASTBUS is taken from the word count word of the list element (LSTWCL/LSTWCH) and not from the data buffer. This option bit is ignored by those elements which do not do single word writes.

4.1.4 OPCHLF - Half Word Transfer

By default, the FSD performs all data transfers between the FASTBUS and UNIBUS in terms of 32-bit words. When this bit is set, data transfers will be done in 16-bit words. When the data is written to FASTBUS, it is right justified in the 32-bit word with the high order 16 bits being set to zero. When the data is read from FASTBUS, the low order 16 bits are written to the UNIBUS and the high order 16 bits are ignored. The data is packed in the UNIBUS buffer (ie. each 16-bit word immediately follows the previous one). This option bit is ignored for FASTBUS to FASTBUS data transfers.

4.1.5 OPCSTD - Standard Opcode

There are two types of opcodes, STANDARD and SPECIAL FUNCTION. Standard opcodes only perform data transfers (UNIBUS to FASTBUS and FASTBUS to FASTBUS). Special function opcodes are used to change internal FSD pointers, FSD parameters, and do primitive FASTBUS operations. When OPCSTD is set, bits 0 through 6 are interpreted as being a standard opcode. When it is clear, these bits are interpreted as being a special function opcode.

4.2 LSTOPT - OPTION BITS

This word contains more option bits as in LSTOPC.

Bit	Name	Description
15	OPCILE	Ignore this element
14	OPCHFB	Hold FASTBUS mastership
13	OPCHFA	Hold FASTBUS address connection
12	OPCHFX	Hold FASTBUS between bursts
11		unused
10		unused
9		unused
8		unused
7		unused
6		unused
5		unused
4		unused
3		unused
2		unused
1	OPCSNR	Suppress null read
0	OPCFIO	FASTBUS device is FIFO like

4.2.1 OPCILE - Ignore The Element

When set, the FSD ignores the function defined by this element. Even though the setting of the bit turns the opcode into a NOOP, bus mastership and a connected device will be released unless option bits OPCHFB and/or OPCHFA are set. A status element is output to the status block indicating this element was skipped.

4.2.2 OPCHFB - Hold FASTBUS Mastership

Normally the FSD releases mastership of the FASTBUS between the execution of list elements. If this bit is set, the FSD will maintain FASTBUS mastership. Setting this bit also implies that OPCHFX is set.

4.2.3 OPCHFA - Hold FASTBUS Address Connection Between Elements

The FSD normally releases any FASTBUS device currently addressed when the list element terminates execution. If OPCHFA is set, the FSD will maintain the address connection between list elements. Setting this bit also impiles that OPCHFB and OPCHFX are set.

4.2.4 OPCHFX - Hold FASTBUS Between Bursts

Normally the FSD releases mastership of FASTBUS between the bursting of data. If OPCHFX is set, the FSD will maintain mastership (and the address lock) between bursts. Please note that even though a slight increase in throughput to the UNIBUS is gained by setting this bit, a major decrease in throughput on FASTBUS in a multi-master environment is possible.

4.2.5 OPCSNR - Suppress Null Read

Whenever an opcode is executed which performs a read block transfer, a null read cycle (read the NTA register) is performed automatically by the FSD to force the slave to remove the signals it is asserting on FASTBUS. If this option bit is set, the null read is suppressed. This feature is usefull if a series of successive read block transfers is performed to the same slave device.

4.2.6 OPCFIO - FASTBUS Device Is FIFO Like

This options bit defines the device on the FASTBUS as being a FIFO (first in, first out) type of device. Certain operations (eg. error retries) are handed differently for FIFO devices than for other FASTBUS devices.

4.3 LSTPAL/LSTPAH - FASTBUS PRIMARY ADDRESS

For those opcodes which do primary address cycles on FASTBUS, these words contain the 32-bit FASTBUS primary address.

4.4 LSTSAL/LSTSAH - FASTBUS SECONDARY ADDRESS

For those opcodes which do secondary address cycles on FASTBUS, these words contain the 32-bit FASTBUS secondary address.

4.5 LSTWCL/LSTWCH - WORD COUNT OR IMMEDIATE DATA

This 32-bit word contains either a word count for block transfers, an offset for special functions, or 32-bits of FASTBUS data for immediate data writes.

CHAPTER 5

STATUS BLOCK FORMAT

The status block is where the FSD reports what actions were taken during the execution of the list. There are two parts to the status block. The status header contains information related to the list as a whole. The status body contains information on an element-by-element basis. When the FSD has completed execution of a list element, a four word status element is written into the next sequential locations in the status body. When the list is terminated, the status header is output. The status block is only guaranteed to be valid when the FSD has completed execution of the list and is in the ready state.

5.1 STATUS HEADER

The initial address of the status block is given by CTLSBA and PRMS17/PRMS16 of CTLPRM. The first ten words of the status block are the header. Such list wide information as the final UNIBUS buffer pointer offset and the connected FASTBUS device address are stored in the header. Table 5-1 shows the format of the status header.

Offset	Name	Description
0	STLESW	Error status word of the last element executed
2	STLCSR	A copy of FSDCSR
4	STLBPL	LSB of the final UNIBUS buffer pointer offset
6	STLBPH	MSB of the final UNIBUS buffer pointer offset
10	STLLPL	LSB of the final list pointer offset
12	STLLPH	MSB of the final list pointer offset
14	STLPAL	LSB of the last FASTBUS primary address
16	STLPAH	MSB of the last FASTBUS primary address
20	STLSAL	LSB of the last FASTBUS secondary address
22	STLSAH	MSB of the last FASTBUS secondary address

Table 5-1. Status header format

5.1.1 STLESW - Error Status Word

The error status word contains flags which indicate what type of errors (or warnings) occurred during the execution of the last list element. This word is a copy of the error status word (STEESW) for the last list element executed prior to list termination. A copy is placed in the header to allow the PDP-11 to quickly determine why the list aborted. See the description of STEESW for a detailed explanation of each bit within this word.

5.1.2 STLCSR - A Copy Of FSDCSR

This word contains a copy of the final contents of FSDCSR. See section 2.8 for a desciption of FSDCSR.

5.1.3 STLBPL/STLBPH - Final UNIBUS Buffer Pointer Offset

This 32-bit word contains the offset in 16-bit words from the start of the UNIBUS buffer to the final buffer pointer.

5.1.4 STLLPL/STLLPH - Final List Pointer Offset

This 32-bit word contains the offset in 16-bit words from the start of the list to the last word plus 1 read by the FSD from the list. The address of the last list element executed is computed by the formula,

initial list address + (2 * (list offset - 8))

The exception to this equation is when the list aborted due to an NXM error while reading the list element. In this case the offset points directly at the element in error (no need to subtract 8).

5.1.5 STLPAL/STLPAH - FASTBUS Primary Address

If CSRHFA is set in FSDCSR, this 32-bit word contains the FASTBUS primary address of the device which was connected to the FSD when the list terminated. If the FSD does not do a secondary address cycle but does do block transfers, the FSD assumes this is a logical address and adds the number of words transferred to the initial primary address. This word will be zero if no FASTBUS device is addressed when the list terminates (ie. no address connection is maintained).

5.1.6 STLSAL/STLSAH - FASTBUS Secondary Address

If CSRHFA is set in FSDCSR, this 32-bit word contains the last secondary address of the device as computed by the FSD. This word is zero if no FASTBUS device is addressed when the list terminates (ie. no address connection is maintained) or if no secondary address cycle was ever performed. The FSD keeps track of the correct secondary address by adding the number of words block transferred to the last secondary address cycle requested. STLSAL/STLSAH reflects this updated secondary address.

5.2 STATUS BODY

Immediately following the status header is the status body. The body is of indeterminate length, since it is based upon the number of list elements executed by the FSD. Each status element written to the body is four words in length. When the FSD has completed execution of the first list element in the list, a status element is written into the body at words 11 through 14. As each list element is executed, the FSD writes the status element in the next sequential memory locations in the status body. All list elements, except for the termination element, have a status element written to the body. The format for a status element is given in table 5-2.

Offset	Name	Description
0	STEESW	Error status word
2	STEISW	Information status word
4	STEWCL	LSB of the word count
6	STEWCH	MSB of the word count

Table 5-2. Status element format

5.2.1 STEESW - Error Status Word

The error status word contains flags which indicate what type of errors (or warnings) occurred during the execution of the list element.

Bit	Name	Description
15	SEEBOV	Buffer overflow
14		unused
13		unused
12	SEEIOC	Illegal opcode
11	SEEFTD	FASTBUS Transfer Device error
10	SEEWPR	Buffer write protected
9	SEEIOP	Illegal operation
8	SEEPER	FASTBUS parity error
7	SEEARB	FASTBUS arbitration timeout
6	SEEDAT	FASTBUS error occurred at data time
5	SEEXAD	FASTBUS error occurred at secondary address time
4	SEEADR	FASTBUS error occurred at address time
3	SEERTO	FASTBUS response timeout
2	SEESS2	FASTBUS slave status bit 2
1	SEESS1	FASTBUS slave status bit 1
0	SEESS0	FASTBUS slave status bit 0

5.2.1.1 SEEBOV - Buffer overflow

The element requested the FSD to transfer more data words than were available in the buffer. Data was transferred until the pointer reached the end of the buffer. The number of words which did get transferred are reported in the word count word of the status element. This error can also occur for special function opcodes which move the internal buffer pointer.

5.2.1.2 SEEIOC - Illegal opcode

There are a total of 256 possible opcodes, not all of which are implemented. An attempt to execute an undefined opcode results in this error.

5.2.1.3 SEEFTD - FASTBUS transfer device error

The error being reported by this status element occured while the FSD was accessing the FASTBUS Transfer Device which was being used as the buffer in a FASTBUS to FASTBUS transfer.

5.2.1.4 SEEWPR - Buffer write protected

The opcode needed to write into the buffer, however the write enable bit (PRMWEN) in CTLPRM was not set. No data was transferred.

5.2.1.5 SEEIOP - Illegal operation

This error occurs when an illegal operation is attempted by a special function opcode. This error is opcode specific.

5.2.1.6 SEEPER - FASTBUS parity error

A FASTBUS parity error occurred during the data transfer. This bit is a warning if parity errors were to be ignored.

5.2.1.7 SEEARB - FASTBUS arbitration timeout

The FSD was unable to win mastership of the FASTBUS.

5.2.1.8 SEEDAT - FASTBUS data time error

The bits SEERTO, SEESS2, SEESS1, and SEESS0 can occur on either primary address, secondary address, or data cycle. If SEEDAT is set, the error reported by these bits occurred on a data cycle.

5.2.1.9 SEEXAD - FASTBUS secondary address error

Same as SEEDAT except the error occurred during a FASTBUS secondary address cycle.

5.2.1.10 SEEADR - FASTBUS primary address error

Same as SEEDAT except the error occurred during a FASTBUS primary address cycle.

5.2.1.11 SEERTO - FASTBUS response timeout

The DK or AK signal did not come back from the slave in an appropriate amount of time. If this is a primary address cycle error, the device most likely does not exist or the SI route tables are not set up correctly. If this is a data cycle error, some type of hardware error has occurred in the connected slave. The address of the device is returned in STLPAL/STLPAH of the status header.

5.2.1.12 SEESS(0,1,2) - FASTBUS slave status response bits

SEESS2, SEESS1 and SEESS0 contain the slave status response which caused the FASTBUS error.

5.2.2 STEISW - Information Status Word

STEISW contains status information about the execution of the list element.

Bit	Name	Description
15	SEEERR	Fatal error indication flag
14	SEEHFB	FASTBUS mastership was held
13	SEEHFA	FASTBUS address connection was held
12	SEERTY	FASTBUS retry was attempted
11		unused
10		unused
9		unused
8		unused
7	SEEWRN	Warning indication flag
6		unused
5		unused
4		unused
3	SEEMDA	Multiple device header status element
2	SEEJSL	CALL/RETURN/JUMP sub-list status element
1	SEENIO	Non-I/O list element
0	SEEILE	Ignored element

5.2.2.1 SEEERR - Error indication flag

This bit is set whenever the element had a fatal error which caused the FSD to abort execution of the list.

5.2.2.2 SEEHFB - FASTBUS mastership was held

The option bit OPCHFB in the list element was set causing the FSD to maintain ownership of the FASTBUS when the element completed.

5.2.2.3 SEEHFA - FASTBUS address connection was held

The option bit OPCHFB in the list element was set causing the FSD to maintain the current address connection when the list element completed.

5.2.2.4 SEERTY - FASTBUS retry was attempted

A FASTBUS error occurred during the execution of this list element and a retry was attempted. If the list was not aborted due to a FASTBUS error the retry was successful. The slave status response which invoked the retry is indicated in STEESW unless the element aborted on a later FASTBUS transfer due to a fatal slave status error.

5.2.2.5 SEEWRN - Warning indication flag

This bit is set if the element had a non-fatal (warning) error. Some warnings are FASTBUS parity errors which were ignored, FASTBUS error retries, or slave status responses which were ignored.

5.2.2.6 SEEMDA - Multiple device header status element

This status element is the header for a multiple device address list element. The parameter word contains the number of device addresses stored in the table on which the opcode will be performed. See chapter 8 for a description of Multiple Device Addressing.

5.2.2.7 SEEJSL - CALL/RETURN/JUMP sub-list status element

The list element which corresponds to this status element was a CALL, RETURN, or JUMP sub-list element (opcode 10, 11, or 12). The parameter word gives the offset in 16-bit words from the start of the list block to the address of the next list element which will be executed.

5.2.2.8 SEENIO - Non-I/O list element

The element did not do any FASTBUS transfers. Only those special function opcodes which do not do any data transfers to/from FASTBUS set this bit.

5.2.2.9 SEEILE - Ignored list element

The list element had its "ignore me" (OPCILE) bit set. The FSD treated the element as a NOOP.

5.2.3 STEWCL/STEWCH - Word Count Or Data

This 32-bit word is opcode specific in its meaning. For all standard opcodes and those special opcodes which do FASTBUS data transfers, this 32-bit word contains the number of data words transferred. If the element terminated in an error, this count does not include the word which caused the error. For the non-I/O special function opcodes, this word is either unused (returned as 0), or contains opcode specific status information, such as offsets. See the specific opcode description for the meaning of this status entry.

CHAPTER 6

OPCODES

The opcode is an 8 bit instruction within the list element which defines the function the FSD is to perform. The opcode is located in bits 0 through 7 of word LSTOPC in the list element. There are 256 possible opcodes which are divided into two classes of 128 opcodes each. The class is selected by bit 7 (OPCSTD) of the opcode. Standard opcodes, selected when OPCSTD is set, perform data transfers on FASTBUS (UNIBUS to FASTBUS or FASTBUS to FASTBUS). Each bit within a standard opcode signifies a FASTBUS operation or control function, such as read vs. write, block vs. single word transfer, etc. These opcodes do all the FASTBUS operations needed to transfer the data, from arbitration to bus release. Special function opcodes, selected when OPCSTD is clear, perform internal FSD operations as well as primitive FASTBUS operations. Not all of the 256 possible opcodes are implemented. An attempt to execute an unimplemented opcode will result in an illegal opcode error which aborts the list.

6.1 STANDARD OPCODES

Standard opcodes (200 through 377) perform complete FASTBUS transactions. They arbitrate, address a device, do an optional secondary address, read or write data in single or block mode, and optionally release the address and bus. Standard options can be decoded bit-wise into specific FASTBUS operation and control, such as addressing control or data space, reading or writing, and single or block data transfers. Table 6-1 outlines how each bit can be decoded.

Bit	Name	Value	Description
7	OPCSTD	1	Must be set for standard opcodes
6	OPCFTD	0	UNIBUS buffer
		1	FASTBUS Transfer Device
5	OPCMDA	0	Single FASTBUS device address
		1	Multiple Device Address
4	OPCXAD	0	No secondary address
		1	secondary address cycle required
3	OPCBLK	0	Single word transfer
		1	Block transfer
2	OPCBRD	0	Single device addressing
		1	Broadcast addressing
1	OPCCTL	0	Address data space
		1	Address control space
0	OPCWRT	0	Write operation
		1	Read operation

Table 6-1. Standard opcode decoding

The following sections describe each standard opcode in detail. Each opcode will transfer in either 16 or 32-bit word mode as specified by the value of OPCHLF in the options word of the element. Those opcodes which do single word writes also check the OPCWID bit for write immediate functions. If OPCWID is set, the single 16 or 32-bit word is taken from the word count word (LSTWCL/LSTWCH) of the list element and not from the buffer. If OPCMDA is set, the opcode is to be applied on the multiple device address and the address contained in LSTPAL/LSTPAH is ignored. If OPCFTD is set, the opcode transfers data to the FASTBUS Transfer Device instead of the UNIBUS buffer. The FASTBUS Transfer Device is defined by special function opcode 060.

If the option bit OPCHFA is set, the device addressed will remain connected to the FSD. Prior to addressing a new device, the FSD will always first release any device left connected to the FSD by this option.

For every opcode, the FASTBUS primary address is given in LSTPAL/LSTPAH (except for multiple device address opcodes); the secondary address is given in LSTSAL/LSTSAH; the word count for block transfers is given in LSTWCL/LSTWCH.

The word count is always given in terms of 16-bit words even if the transfer mode is 32-bit. For instance, if it is desired to write 256 32-bit words to a device, the word count would be 512. If an odd count is given while in 32-bit mode, the count is rounded down (a word count of 11 would transfer 5 32-bit words). FASTBUS to FASTBUS transfers also require the word count to be in terms of 16-bit words.

The list element format for each standard opcode is defined below. The offset is the number of bytes from the current list element pointer.

Offset)	contents
2, 0)	option//opcode
6, 4)	FASTBUS primary address
12,10)	FASTBUS secondary address
16,14)	Word count or immediate data

The status element word always contains the number of 16-bit words read or written by the FSD. The possible errors which can occur during the execution of a standard opcode are:

Possible errors: SEEBOV, SEEWPR, SEEPER
SEEARB, SEERTO, SEESS(0,1,2)

The following table lists the standard opcodes which perform UNIBUS to FASTBUS transfers with single device addressing.

Opcode	R/W	Space	Block	Broadcast	Secondary
200	W	data	no	no	no
201	R	data	no	no	no
202	W	control	no	no	no
203	R	control	no	no	no
204	W	data	no	yes	no
205	R	data	no	yes	no
206	W	control	no	yes	no
207	R	control	no	yes	no
210	W	data	yes	no	no
211	R	data	yes	no	no
212	W	control	yes	no	no
213	R	control	yes	no	no
214	W	data	yes	yes	no
215	R	data	yes	yes	no
216	W	control	yes	yes	no
217	R	control	yes	yes	no
220	W	data	no	no	yes
221	R	data	no	no	yes
222	W	control	no	no	yes
223	R	control	no	no	yes
224	W	data	no	yes	yes
225	R	data	no	yes	yes
226	W	control	no	yes	yes
227	R	control	no	yes	yes
230	W	data	yes	no	yes
231	R	data	yes	no	yes
232	W	control	yes	no	yes
233	R	control	yes	no	yes
234	W	data	yes	yes	yes
235	R	data	yes	yes	yes
236	W	control	yes	yes	yes
237	R	control	yes	yes	yes

Table 6-2. Single address opcodes for UNIBUS transfers

The following table lists the standard opcodes which perform UNIBUS to FASTBUS transfers with multiple device addressing.

Opcode	R/W	Space	Block	Broadcast	Secondary
240	W	data	no	no	no
241	R	data	no	no	no
242	W	control	no	no	no
243	R	control	no	no	no
244	W	data	no	yes	no
245	R	data	no	yes	no
246	W	control	no	yes	no
247	R	control	no	yes	no
250	W	data	yes	no	no
251	R	data	yes	no	no
252	W	control	yes	no	no
253	R	control	yes	no	no
254	W	data	yes	yes	no
255	R	data	yes	yes	no
256	W	control	yes	yes	no
257	R	control	yes	yes	no
260	W	data	no	no	yes
261	R	data	no	no	yes
262	W	control	no	no	yes
263	R	control	no	no	yes
264	W	data	no	yes	yes
265	R	data	no	yes	yes
266	W	control	no	yes	yes
267	R	control	no	yes	yes
270	W	data	yes	no	yes
271	R	data	yes	no	yes
272	W	control	yes	no	yes
273	R	control	yes	no	yes
274	W	data	yes	yes	yes
275	R	data	yes	yes	yes
276	W	control	yes	yes	yes
277	R	control	yes	yes	yes

Table 6-3. Multiple address opcodes for UNIBUS transfers

The following table lists the standard opcodes which perform FASTBUS to FASTBUS transfers with single device addressing.

Opcode	R/W	Space	Block	Broadcast	Secondary
300	W	data	no	no	no
301	R	data	no	no	no
302	W	control	no	no	no
303	R	control	no	no	no
304	W	data	no	yes	no
305	R	data	no	yes	no
306	W	control	no	yes	no
307	R	control	no	yes	no
310	W	data	yes	no	no
311	R	data	yes	no	no
312	W	control	yes	no	no
313	R	control	yes	no	no
314	W	data	yes	yes	no
315	R	data	yes	yes	no
316	W	control	yes	yes	no
317	R	control	yes	yes	no
320	W	data	no	no	yes
321	R	data	no	no	yes
322	W	control	no	no	yes
323	R	control	no	no	yes
324	W	data	no	yes	yes
325	R	data	no	yes	yes
326	W	control	no	yes	yes
327	R	control	no	yes	yes
330	W	data	yes	no	yes
331	R	data	yes	no	yes
332	W	control	yes	no	yes
333	R	control	yes	no	yes
334	W	data	yes	yes	yes
335	R	data	yes	yes	yes
336	W	control	yes	yes	yes
337	R	control	yes	yes	yes

Table 6-4. Single address opcodes for FASTBUS transfers

The following table lists the standard opcodes which perform FASTBUS to FASTBUS transfers with multiple device addressing.

Opcode	R/W	Space	Block	Broadcast	Secondary
340	W	data	no	no	no
341	R	data	no	no	no
342	W	control	no	no	no
343	R	control	no	no	no
344	W	data	no	yes	no
345	R	data	no	yes	no
346	W	control	no	yes	no
347	R	control	no	yes	no
350	W	data	yes	no	no
351	R	data	yes	no	no
352	W	control	yes	no	no
353	R	control	yes	no	no
354	W	data	yes	yes	no
355	R	data	yes	yes	no
356	W	control	yes	yes	no
357	R	control	yes	yes	no
360	W	data	no	no	yes
361	R	data	no	no	yes
362	W	control	no	no	yes
363	R	control	no	no	yes
364	W	data	no	yes	yes
365	R	data	no	yes	yes
366	W	control	no	yes	yes
367	R	control	no	yes	yes
370	W	data	yes	no	yes
371	R	data	yes	no	yes
372	W	control	yes	no	yes
373	R	control	yes	no	yes
374	W	data	yes	yes	yes
375	R	data	yes	yes	yes
376	W	control	yes	yes	yes
377	R	control	yes	yes	yes

Table 6-5. Multiple address opcodes for FASTBUS transfers

6.2 OPCODES 200 THROUGH 277

The following sections (6.2.1 through 6.2.32) describe the standard function opcodes 200 through 277. These opcodes transfer data between the FASTBUS and the UNIBUS buffer. Only one FASTBUS device is addressed per list element.

6.2.1 Opcode 200 - Single Write Data Space

The FASTBUS device is addressed in data space and a single word, taken from either the buffer or list element, is written to the device.

6.2.2 Opcode 201 - Single Read Data Space

The FASTBUS device is addressed in data space and a single word is read from the device into the buffer.

6.2.3 Opcode 202 - Single Write Control Space

Same as opcode 200 except the device is addressed in control space.

6.2.4 Opcode 203 - Single Read Control Space

Same as opcode 201 except the device is addressed in control space.

6.2.5 Opcode 204 - Broadcast Single Write Data Space

A broadcast address cycle is performed to data space. This is followed by a write of a single word take from the buffer or list element.

6.2.6 Opcode 205 - Broadcast Single Read Data Space

A broadcast address cycle is performed to data space. This is followed by a read of a single word into the buffer.

6.2.7 Opcode 206 - Broadcast Write Single Word Control Space

Same as opcode 204 except control space is addressed.

6.2.8 Opcode 207 - Broadcast Single Read Control Space

Same as opcode 205 except control space is addressed.

6.2.9 Opcode 210 - Block Write Data Space

The FASTBUS device is addressed in data space and a block of words taken from the buffer is written to the device.

6.2.10 Opcode 211 - Block Read Data Space

The FASTBUS device is addressed in data space and a block of words is read from the device into the buffer.

6.2.11 Opcode 212 - Block Write Control Space

Same as opcode 210 except control space is addressed.

6.2.12 Opcode 213 - Block Read Control Space

Same as opcode 211 except control space is addressed.

6.2.13 Opcode 214 - Broadcast Block Write Data Space

A broadcast address cycle is performed to data space. This is followed by a block write of data from the buffer.

6.2.14 Opcode 215 - Broadcast Block Read Data Space

A broadcast address cycle is performed to data space. This is followed by a block read of data into the buffer. The interpretation of the data from this opcode is unclear.

6.2.15 Opcode 216 - Broadcast Block Write Control Space

Same as opcode 214 except control space is addressed.

6.2.16 Opcode 217 - Broadcast Block Read Control Space

Same as opcode 215 except control space is addressed

6.2.17 Opcode 220 - Secondary Single Write Data Space

The FASTBUS device is addressed in data space. The address cycle is followed by a secondary address cycle and a single word write cycle.

6.2.18 Opcode 221 - Secondary Single Read Data Space

The FASTBUS device is addressed in data space. The address cycle is followed by a secondary address cycle and a single word read cycle.

6.2.19 Opcode 222 - Secondary Single Write Control Space

Same as opcode 220 except control space is addressed

6.2.20 Opcode 223 - Secondary Single Read Control Space

Same as opcode 221 except control space is addressed.

6.2.21 Opcode 224 - Broadcast Secondary Single Write Data Space

A broadcast address cycle is performed to data space. The address cycle is followed by a secondary address cycle and a single write cycle.

6.2.22 Opcode 225 - Broadcast Secondary Single Read Data Space

A broadcast address cycle is performed to data space. The address cycle is followed by a secondary address cycle and a single read cycle.

6.2.23 Opcode 226 - Broadcast Secondary Single Write Control Space

Same as opcode 224 except control space is addressed.

6.2.24 Opcode 227 - Broadcast Secondary Single Read Control Space

Same as opcode 225 except control space is addressed

6.2.25 Opcode 230 - Secondary Block Write Data Space

The FASTBUS device is addressed in data space. This address cycle is followed by a secondary address cycle and a block write.

6.2.26 Opcode 231 - Secondary Block Read Data Space

The FASTBUS device is addressed in data space. This address cycle is followed by a secondary address cycle and a block read.

6.2.27 Opcode 232 - Secondary Block Write Control Space

Same as opcode 230 except control space is addressed.

6.2.28 Opcode 233 - Secondary Block Read Control Space

Same as opcode 231 except control space is addressed.

6.2.29 Opcode 234 - Broadcast Secondary Block Write Data Space

A broadcast address cycle is performed to data space. This address cycle is followed by a secondary address cycle and a block write.

6.2.30 Opcode 235 - Broadcast Secondary Block Read Data Space

A broadcast address cycle is performed to data space. This address cycle is followed by a secondary address cycle and a block read. The result of this opcode is unclear.

6.2.31 Opcode 236 - Broadcast Secondary Block Write Control Space

Same as opcode 234 except control space is addressed.

6.2.32 Opcode 237 - Broadcast Secondary Block Read Control Space

Same as opcode 235 except control space is addressed.

6.3 OPCODES 240 THROUGH 277 - MULTIPLE DEVICE ADDRESS OPCODES

Opcodes 240 through 277 perform the same functions as opcodes 200 through 237. The extension is that the function is performed on a list of FASTBUS device addresses. These addresses are fetched from the Multiple Device Address table contained within the FSD. This table is loaded by a special function opcode. The task specified by the opcode is performed once for each address contained within the table. The same result could be achieved if the list contained an element for every address within the MDA table. Such opcodes are useful for performing identical functions on a large group of FASTBUS devices.

For each address contained in the MDA table, a separate status element is written into the status body. The programmer should take this into consideration when allocating a buffer for the status block. The status element is of the same form as for the other standard opcodes. In the present microcode, 63 addresses can be loaded into the table at one time.

6.4 OPCODES 300 THROUGH 377 - FASTBUS TO FASTBUS TRANSFER

Opcodes 300 through 377 perform the same functions as opcodes 200 through 277 except that the data is never transferred to or from the UNIBUS buffer. The data buffer actually resides in what is referred to as the FASTBUS Transfer Device. This device, which in most cases will be a random access memory, acts as the buffer for the transfer. This FASTBUS to FASTBUS transfer is very high speed since no UNIBUS access is required.

6.5 SPECIAL OPCODES

Special opcodes (000 through 177) perform internal FSD operations as well as primitive FASTBUS operations. The opcode for a special function cannot be decoded as the standard opcodes. Also the use of the parmameter words in the list element are opcode specific and cannot be generalized. The use of a special opcode automatically implies the FASTBUS is not to be released when the opcode is complete. This is the same as setting bit OPCHFB in the options word of the element. Each opcode description includes the parameter word usage, possible errors, and the status element parameter word which will be written.

For example:

Status element: 16-bit word offset

Possible errors: SEEBOV, SEEIOP

2, 0) option//opcode
6, 4) FASTBUS primary address
12,10) FASTBUS secondary address
14,16) Word count, offset, data, etc

Opcode	Description
000	Termination element
001	Diagnostic, dump internal FSD ram
002	Diagnostic, dump internal FSD registers
003	Move buffer address pointer
004	Set buffer address pointer
005	Save current buffer address pointer
006	Restore buffer address pointer
007	Save a computed word count
010	Call a sub-list
011	Return from a sub-list
012	Jump to a new list
013	Returns the version/edit of microcode
014	Set the multiple device address table
015	Read the multiple device address table
016	Set the system parameter block
017	Read the system parameter block
020	Set the burst size
021	Set the clock cycle (not used)
022	Set the retry counter
023	Set the arbitration vector (not used)
024	Set the primary address error response word
025	Set the secondary address error response word
026	Set the data error response word
027	reserved
030	List element interrupt
031	Arbitrate for the FASTBUS
032	Release FASTBUS mastership
033	Release FASTBUS address lock
034	Address FASTBUS in data space
035	Address FASTBUS in control space
036	Broadcast address FASTBUS in data space
037	Broadcast address FASTBUS in control space
040	UNIBUS transfer, write a single word of data
041	UNIBUS transfer, write a block of data
042	UNIBUS transfer, write a secondary address
043	reserved
044	FASTBUS transfer, write a single data word
045	FASTBUS transfer, write a block of data
046	FASTBUS transfer, write a secondary address
047	reserved
050	UNIBUS transfer, read a single data word
051	UNIBUS transfer, read a block of data
052	UNIBUS transfer, read a secondary address
053	reserved
054	FASTBUS transfer, read a single word of data
055	FASTBUS transfer, read a block of data
056	FASTBUS transfer, read a secondary address
057	reserved
060	Set the FASTBUS Transfer Device address
061	Read the FASTBUS Transfer Device address

Table 6-6. Special function opcodes

6.5.1 Opcode 000 - Termination Element

The termination element causes the FSD to end the execution of the list. This is the only element which does not have a corresponding status element written into the status block. Any device connected to the FSD remains connected. If the FSD is bus master, it retains control of the bus. The status header is output to the status block and the FSD is marked "ready" in register FSDHCS.

Status element: None written

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

6.5.2 Opcode 001 - Diagnostic Opcode, Dump FSD Ram

This is a diagnostic opcode and is not intended for general use. This opcode returns the contents of the 1K internal FSD memory. The parameters are the starting address within the ram (0 to 17777 octal) and the number of words to dump. The data, which is always 32-bit words, is returned to the buffer. A buffer overflow error will occur if there is not enough room in the buffer. If the number of words to dump goes beyond the 1K length, an illegal operation error occurs.

Status element: number of 16-bit words dumped

Possible errors: SEEBOV, SEEIOP, SEEWPR

```
 2, 0) option//opcode
 6, 4) starting ram address
12,10) unused
16,14) number of 16-bit words to dump
```

6.5.3 Opcode 002 - Diagnostic Opcode, Dump FSD Registers

This is a diagnostic opcode and is not intended for general use. This opcode returns the contents of the seventeen 32-bit internal FSD registers. No parameters are needed for this opcode. The contents of the registers are returned to the buffer. A buffer overflow error will occur if there is not enough room in the buffer.

Status element: number of 16-bit words dumped

Possible errors: SEEBOV, SEEWPR

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

6.5.4 Opcode 003 - Move The UNIBUS Buffer Address Pointer

The UNIBUS buffer pointer is moved by the offset given in the element. The offset is the number of 16-bit words to move the pointer from its current location. It can be a negative number (2's complement). An illegal operation error will result if the new pointer is lower than the initial UNIBUS buffer address or beyond the defined buffer length.

Status element: 16-bit word offset the pointer was moved

Possible errors: SEEIOP

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) offset in 16-bit words to move the pointer
```

6.5.5 Opcode 004 - Set The UNIBUS Buffer Address Pointer

This opcode is similar to 003 except the offset is from the initial UNIBUS buffer address given in the control block. For example, to set the pointer to the initial address, the offset would be 0. An illegal operation error will result if the pointer is set beyond the defined bounds of the buffer.

Status element: 16-bit word offset the pointer was moved

Possible errors: SEEIOP

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) offset in 16-bit words to set the pointer
```

6.5.6 Opcode 005 - Save The UNIBUS Buffer Address Pointer

The current UNIBUS buffer address pointer is saved on an internal FSD stack. A maximum of 15 pointers can be placed on the stack. An illegal operation error will result if the stack is full. The buffer pointer can also be incremented at the same time to leave room for a word count word (see opcode 007).

Status element: number of 16-bit words the pointer was moved

Possible errors: SEEIOP

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) offset in 16-bit words to move the pointer
```

6.5.7 Opcode 006 - Restore The UNIBUS Buffer Address Pointer

A UNIBUS buffer address pointer, previously saved on the internal stack by opcode 005, is restored as the current pointer. An illegal operation error will result if the stack is empty.

Status element: zero

Possible errors: SEEIOP

2, 0) option//opcode
6, 4) unused
12,10) unused
16,14) unused

6.5.8 Opcode 007 - Save A Computed Word Count

This opcode is used together with opcode 005 to compute the number of 16-bit words stored in the buffer. The last address saved on the internal stack is popped and subtracted from the current UNIBUS buffer address pointer. This number is stored in the UNIBUS buffer at the saved address. The current buffer address pointer remains unchanged. An illegal operation error will result if the stack is empty. A usefull sequence would be opcode 005 with an offset of 2 (1 if 16-bit transfer mode), FASTBUS reads, and ending with this opcode. The number of 16-bit words read from FASTBUS would be written to the saved buffer location.

Status element: number of 16-bit words stored in the buffer

Possible errors: SEEIOP

2, 0) option//opcode
6, 4) unused
12,10) unused
16,14) unused

6.5.9 Opcode 010 - Call A Sub-list

This opcode provides the means to create subroutine lists which can be called by the main list. The starting address of the sub-list is passed as an offset in 16-bit words from the initial list address. The offset can be a negative number. The address of the next list element is saved on an internal stack up to a depth of 15. An illegal operation error will result if this limit is exceeded.

Status element: new UNIBUS list address

Possible errors: SEEIOP

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) offset in 16-bit words from initial list address
```

6.5.10 Opcode 011 - Return From A Sub-list Call

This is the complement of opcode 010. The list address saved on the stack by opcode 010 is popped and used as the address for the next list element. An illegal operation error will result if the stack is empty.

Status element: saved UNIBUS list address

Possible errors: SEEIOP

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

6.5.11 Opcode 012 - Jump To A New List

This opcode is similar to opcode 010 except the current list address is not saved on the stack. The address of the new list is passed as an offset in 16-bit words from the initial list address and can be negative. There are no errors associated with this element.

Status element: new UNIBUS list address

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) offset in 16-bit words from initial list address
```

6.5.12 Opcode 013 - Return The Version Of The Microcode

The version, release level and edit level of the microcode is returned in the status element. The edit number is contained in bits 0 through 15, the release level in bits 16 through 23, and the version in bits 24 through 31.

Status element: version/release/edit number

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

6.5.13 Opcode 014 - Set The Multiple Device Address Table

The Multiple Device Address table is used by standard opcodes 240 through 277 and 340 through 377. This opcode loads the table from the UNIBUS buffer into memory internal to the FSD. Each primary address in the table is 32-bits. A maximum of 63 primary addresses can be loaded into the table. If the number is 0, the current table is cleared. Any attempt to execute a MDA opcode with a cleared table will result in an illegal operation error. The MDA table remains loaded between lists. It is only cleared during a powerup or hardware reset.

Status element: number of primary addresses stored

Possible errors: SEEBOV, SEEIOP

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) number of primary addresses to load
```

6.5.14 Opcode 015 - Read The Multiple Device Address Table

The Multiple Device Address table is copied to the UNIBUS buffer. The total number of primary addresses returned is stored in the status element. Each primary address is a 32-bit word.

Status element: number of primary addresses stored

Possible errors: SEEBOV, SEEWPR

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

6.5.15 Opcode 016 - Set The System Parameter Block

The system parameter block is a block of eight 32-bit words which contains parameters for the operation of the FSD. The parameter block is read from the UNIBUS buffer and stored in memory internal to the FSD. Not all entries in the parameter block are used by the FSD. However all locations must be allocated in the buffer. The contents of these locations are ignored, but are stored within the FSD as given. Opcode 017 which reads this table will return the contents as stored. The system parameter block remains loaded between lists. It is only cleared during a powerup or hardware reset. The following table outlines the contents of the system parameter block. Each entry in the system parameter block can be set individually by other special function opcodes.

Offset	Name	Description
0	SYPBRS	Burst size for block transfers
4	SYPCLK	Clock cycle (unused)
10	SYPRTY	Retry counter
14	SYPARB	Arbitration vector (unused)
20	SYPAEC	Primary address error control word
24	SYPXEC	Secondary address error control word
30	SYPDEC	Data error control word
34		reserved for future assignment

Status element: always contains 16

Possible errors: SEEBOV, SEEIOP

2, 0) option//opcode
6, 4) unused
12,10) unused
16,14) unused

6.5.16 Opcode 017 - Read The System Parameter Block

This opcode will read the system parameter block into the UNIBUS buffer. The system parameter block is outlined in opcode 016.

Status element: always contains 16

Possible errors: SEEBOV, SEEWPR

2, 0) option//opcode
6, 4) unused
12,10) unused
16,14) unused

6.5.17 Opcode 020 - Set The Burst Size

Due to hardware restrictions, there is a maximum limit as to the number of words which can be transferred at one time on the FASTBUS. Block transfer requests greater than this maximum are broken down into several bursts whose length is governed by the burst size. The burst size is independent of the transfer mode size (16 or 32-bit). The hardware maximum for a burst is 256 words. The default setting for the burst size is 256. This opcode allows the burst size to be set to any value between 1 and 256. If the new burst size exceed these boundaries, the maximum burst size is used. Since data is transferred between FASTBUS and the UNIBUS through a memory buffer internal to the FSD, the larger the burst size the less overhead involved in the transfer.

Status element: new burst size

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) new burst size
```

6.5.18 Opcode 021 - Set The Clock Cycle

This opcode will change the clock cycle entry in the system parameter block. The clock cycle is not used by the FSD and the contents of this word are ignored. Opcode 017 (read system parameter block) will return the contents of this word.

Status element: new clock cycle

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) new clock cycle
```

6.5.19 Opcode 022 - Set The Retry Counter

FASTBUS errors can be automatically retried by the FSD. The retry counter is the number of times the FSD will attempt to retry the same FASTBUS cycle before giving up with a fatal error. The default retry counter is 5 and can be set to any number between 0 and 2**18. If the new count exceeds these boundaries, the maximum value is used.

Status element: new retry counter

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) new retry counter
```

6.5.20 Opcode 023 - Set The Arbitration Vector

This opcode will change the arbitration vector entry in the system parameter block. The arbitration vector cannot be changed by the FSD microcode. It can only be modified by the PDP-11 through the I/O-page register or by accessing CSR 8 from FASTBUS. The entry for the arbitration vector is therefore ignored but is stored unmodified in the FSD. Opcode 017 (read system parameter block) will return the contents of this word.

Status element: new arbitration vector

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) new arbitration vector
```

6.5.21 Opcode 024 - Set The Primary Address Error Response Word

This opcode sets the primary address error response word for the FSD.

Status element: primary address error response word

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) primary address error response word
```

6.5.22 Opcode 025 - Set The Secondary Address Error Response Word

This opcode sets the secondary address error response word for the FSD.

Status element: secondary address error response word

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) secondary address error response word
```

6.5.23 Opcode 026 - Set The Data Error Response Word

This opcode sets the data error response word for the FSD.

Status element: data error response word

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) data error response word
```

6.5.24 Opcode 030 - List Element Interrupt

This opcode emulates the setting of the halt bit (HCSHLT) in FSDHCS for the FSD. After the execution of the element, the halt bit in FSDHCS is set to one, the FSD becomes ready (generating an interrupt if HCSIE is also set), and the FSD waits until either the halt bit is cleared or the execution bit (HCSXCT) is set. This opcode is useful for read/modify/write type operations or conditional execution of the remaining portion of the list based on data read in by the portion of the list preceeding this element.

Status element: zero

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

6.5.25 Opcode 031 - Arbitrate For FASTBUS

This opcode allows the PDP-11 to arbitrate for the FASTBUS. If the FSD is already FASTBUS master, the bus and any address are released prior to the new arbitration cycle. Arbitrating for the FASTBUS is not required when doing the primitive opcodes since the FSD will automatically arbitrate whenever a FASTBUS primary address cycle is to be performed and it is not master of the FASTBUS.

Status element: zero

Possible errors: SEEARB

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

6.5.26 Opcode 032 - Release FASTBUS Mastership

This opcode will make the FSD release any address lock and mastership of FASTBUS. If the FSD is not master of FASTBUS, the opcode is a NOOP.

Status element: zero

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

6.5.27 Opcode 033 - Release The FASTBUS Address Lock

The FASTBUS device currently addressed is released however FASTBUS mastership is retained. If there is no device currently addressed the opcode is a NOOP.

Status element: zero

Possible errors: None

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

6.5.28 Opcodes 034,035,036,037 - FASTBUS Primary Address Cycles

These four opcodes perform the four different primary address cycles available on FASTBUS. For all four, the FASTBUS primary address is taken from the primary address word of the list element. If the FSD is not FASTBUS master, an arbitration cycle is done prior to the primary address cycle. If a device is already addressed, it is released prior to the new address cycle.

The four possible primary address cycles are:

034	Address	data space
035	Address	control space
036	Broadcast	data space
037	Broadcast	control space

Status element: zero

Possible errors: SEEARB, SEERTO, SEESS(0,1,2)

2, 0)	option//opcode
6, 4)	FASTBUS primary address
12,10)	unused
16,14)	unused

6.5.29 Opcode 040 - UNIBUS Transfer, Single Word Write Of Data

A single data word is written to the FASTBUS device currently addressed. The data can come from either the UNIBUS buffer or the list element depending on the setting of bit OPCWID in the options word. The data word written is either 16 or 32-bits depending on the setting of the OPCHLF bit in the options word. Should the FSD not be connected to a device, the list will abort with an illegal operation error.

Status element: number of 16-bit words written (0, 1, or 2)

Possible errors: SEEBOV, SEEIOP, SEERTO, SEESS(0,1,2)

2, 0)	option//opcode
6, 4)	unused
12,10)	unused
16,14)	immediate data if OPCWID=1

6.5.30 Opcode 041 - UNIBUS Transfer, Block Write Of Data

A block of data words taken from the UNIBUS buffer is written to the FASTBUS device currently connected by the FSD. If no device is connected an illegal operation error will abort the list. The transfer will be either 16 or 32-bit mode depending on the setting of the OPCHLF bit in the options word.

Status element: number of 16-bit words written

Possible errors: SEEBOV, SEEIOP, SEERTO, SEESS(0,1,2)

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) word count
```

6.5.31 Opcode 042 - UNIBUS Transfer, Write A Secondary Address

Opcode 042 does a secondary address cycle to the device currently addressed. If no device is connected, the list is terminated in an illegal operation error. The secondary address is taken from the secondary address word of the list element.

Status element: zero

Possible errors: SEEIOP, SEERTO, SEESS(0,1,2)

```
 2, 0) option//opcode
 6, 4) unused
12,10) FASTBUS secondary address
16,14) unused
```

6.5.32 Opcode 044 - FASTBUS Transfer, Single Word Write Of Data

A single data word is written to the FASTBUS device currently addressed. The data can come from either the FASTBUS transfer device or the list element depending on the setting of bit OPCWID in the options word. The data word written is either 16 or 32-bits depending on the setting of the OPCHLF bit in the options word. Should the FSD not be connected to a device, the list will abort with an illegal operation error.

Status element: number of 16-bit words written (0, 1, or 2)

Possible errors: SEEBOV, SEEIOP, SEERTO, SEESS(0,1,2)

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) immediate data if OPCWID=1
```

6.5.33 Opcode 045 - FASTBUS Transfer, Block Write Of Data

A block of data words taken from the FASTBUS transfer device is written to the FASTBUS device currently connected by the FSD. If no device is connected an illegal operation error will abort the list. The transfer will be either 16 or 32-bit mode depending on the setting of the OPCHLF bit in the options word.

Status element: number of 16-bit words written

Possible errors: SEEBOV, SEEIOP, SEERTO, SEESS(0,1,2)

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) word count
```

6.5.34 Opcode 046 - FASTBUS Transfer, Write A Secondary Address

Opcode 044 does a secondary address cycle to the device currently addressed. If no device is connected, the list is terminated in an illegal operation error. The secondary address is taken from the secondary address word of the list element. This opcode performs the same function as opcode 042.

Status element: zero

Possible errors: SEEIOP, SEERTO, SEESS(0,1,2)

```
 2, 0) option//opcode
 6, 4) unused
12,10) FASTBUS secondary address
16,14) unused
```

6.5.35 Opcode 050 - UNIBUS Transfer, Single Word Read Of Data

A single data word is read from the FASTBUS device currently addressed and stored in the UNIBUS buffer. The data word read is either 16 or 32-bits depending on the setting of the OPCHLF bit in the options word. Should the FSD not be connected to a device, the list will abort with an illegal operation error.

Status element: number of 16-bit words read (0, 1, or 2)

Possible errors: SEEBOV, SEEIOP, SEEWPR, SEEPER
SEERTO, SEESS(0,1,2)

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

6.5.36 Opcode 051 - UNIBUS Transfer, Block Read Of Data

A block of data words is read from the FASTBUS device currently connected and written into the UNIBUS buffer. If no device is connected an illegal operation error will abort the list. The transfer will be either 16 or 32-bit mode depending on the setting of the OPCHLF bit in the options word.

Status element: number of 16-bit words read

Possible errors: SEEBOV, SEEIOP, SEEWPR, SEEPER
SEERTO, SEESS(0,1,2)

2, 0) option//opcode
6, 4) unused
12,10) unused
16,14) unused

6.5.37 Opcode 052 - UNIBUS Transfer, Read A Secondary Address

This opcode performs a secondary address read cycle on the device connected. If no device is connected, the list will abort with an illegal operation error. This type of cycle will return the current value of the NTA register within the device. The data returned by the device is always stored in the UNIBUS buffer as a 32-bit word.

Status element: number of 16-bit words read (0 or 2)

Possible errors: SEEBOV, SEEIOP, SEEWPR, SEEPER
SEERTO, SEESS(0,1,2)

2, 0) option//opcode
6, 4) unused
12,10) unused
16,14) unused

6.5.38 Opcode 054 - FASTBUS Transfer, Single Word Read Of Data

A single data word is read from the FASTBUS device currently addressed and stored in the FASTBUS transfer device. The data word read is either 16 or 32-bits depending on the setting of the OPCHLF bit in the options word. Should the FSD not be connected to a device, the list will abort with an illegal operation error.

Status element: number of 16-bit words read (0, 1, or 2)

Possible errors: SEEBOV, SEEIOP, SEEPER
SEERTO, SEESS(0,1,2)

2, 0) option//opcode
6, 4) unused
12,10) unused
16,14) unused

6.5.39 Opcode 055 - FASTBUS Transfer, Block Read Of Data

A block of data words is read from the FASTBUS device currently connected and written into the FASTBUS transfer device. If no device is connected an illegal operation error will abort the list. The transfer will be either 16 or 32-bit mode depending on the setting of the OPCHLF bit in the options word.

Status element: number of 16-bit words read

Possible errors: SEEBOV, SEEIOP, SEEPER
SEERTO, SEESS(0,1,2)

2, 0) option//opcode
6, 4) unused
12,10) unused
16,14) unused

6.5.40 Opcode 056 - FASTBUS Transfer, Read A Secondary Address

This opcode performs a secondary address read cycle on the device connected. If no device is connected, the list will abort with an illegal operation error. This type of cycle will return the current value of the NTA register within the device. The data returned by the device is always stored in the FASTBUS transfer device as a 32-bit word.

Status element: number of 16-bit words read (0 or 2)

Possible errors: SEEBOV, SEEIOP, SEEPER
SEERTO, SEESS(0,1,2)

2, 0) option//opcode
6, 4) unused
12,10) unused
16,14) unused

6.5.41 Opcode 060 - Set The FASTBUS Transfer Device Address

This opcode sets the FASTBUS Transfer Device address which is used by standard opcodes 300 through 377 and special function opcodes 44, 45, 46, 54, 55 and 56. The list element contains the primary address, the secondary address if needed, and the total number of words that can be stored in the device. If OPCFTS is clear, only a primary address is done by the FSD when the FASTBUS Transfer Device is addressed. Should OPCFTS be set, the FSD will do a secondary address after the primary address connection to the device.

Status element: zero

Possible errors: none

2, 0) option//opcode
6, 4) FASTBUS primary address
12,10) secondary address (optional)
16,14) maximum transfer word count

6.5.42 Opcode 061 - Read The FASTBUS Transfer Device Address

This opcode causes the FSD to write the current information it contains on the FASTBUS Transfer Device. Five 32-bit words are returned to the UNIBUS buffer. The first four words returned are the list element which was used to load the FASTBUS Transfer Device. The fifth word contains the total number of words transferred to this device since it was defined.

Status element: always contains 5

Possible errors: SEEBOV, SEEWPR

```
 2, 0) option//opcode
 6, 4) unused
12,10) unused
16,14) unused
```

CHAPTER 7

ERROR RESPONSES

During the course of operations, FASTBUS errors are likely to occur. In some circumstances the user may want to ignore the error. In other cases a retry might be in order. In others, the error could be considered fatal. To help speed up total throughput between the UNIBUS and FASTBUS, the FSD is able to handle errors without the PDP-11 intervening. The user programs the FSD error control through the error response words (ERW). These words, which are part of the system parameter block, contain information, the error response code, which informs the FSD how to handle non-zero slave status response and response timeouts. The three basic FASTBUS cycles (primary addressing, secondary addressing and data) each have their own error response word. Parity error responses can also be programmed. Their error response code is contained in the list element options word.

When an error occurs, the FSD looks in the appropriate location for control information. For response timeouts and slave status errors, the FSD refers to the system parameter block error response words. For parity errors, the FSD looks at the list element. The error response code (ERC) is decoded and the FSD takes the appropriate action based upon this code.

7.1 ERROR RESPONSE WORDS

Error response words (ERW) are 32-bit words contained in the system parameter block which contain the information the FSD needs for handling FASTBUS errors. The ERW controls the FSD's response to non-zero slave status responses (SS=1 through SS=7) and response timeouts (no AK or DK). The ERW is broken into eight 4-bit fields. The first field, in bits 0-3, contains the error response code (ERC) for response timeouts. The remaining fields contain the ERC for SS responses 1 through 7.

There are three different error response words, one for each type of FASTBUS cycle. The primary address ERW controls all errors which occur during primary address cycles. The secondary address ERW controls all errors which occur during secondary address cycles. The data ERW controls all errors which occur during data cycle, both single and block mode transfers. The FSD remembers the setting of the ERWs between lists. They are only changed by either loading a new system parameter block or by executing the list elements which specifically change the error response words. The default setting for all the fields in the error response words is such that all errors are treated as fatal errors.

Most significant bits							
15 12	11 8	7 4	3 0	15 12	11 8	7 4	3 0
0 3	4 7	8 11	12 15	16 19	20 23	24 27	28 31
RTO	SS=1	SS=2	SS=3	SS=4	SS=5	SS=6	SS=7

Table 7-1. Error response word layout

7.2 PARITY ERROR RESPONSE

Parity errors, which can only be detected when the FSD performs FASTBUS read cycles, are controled on a list element basis. Unlike response timeout and slave status errors, the error response code for parity errors is contained within the list element performing the FASTBUS read cycle. The ERC resides in the list element options word, bits 16 through 19 (bits 12 through 15 of LSTOPC).

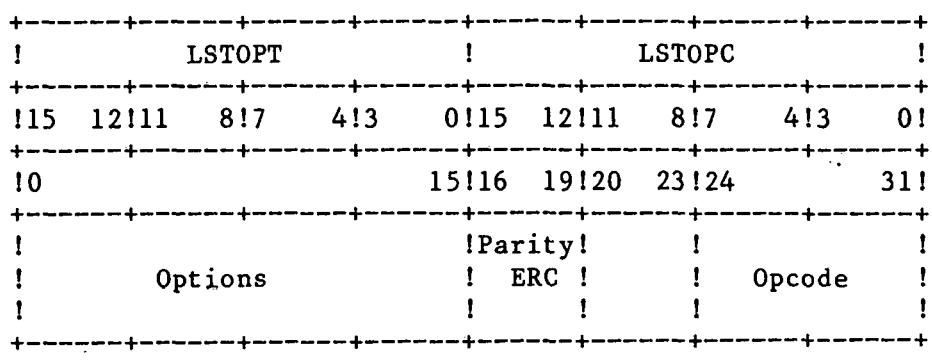

Table 7-2. List element options word layout

7.3 ERROR RESPONSE CODES

There are 16 possible error response codes (ERC 0 through 15), however the FSD currently only uses seven of these. The seven different error response codes are:

ERC	Description
0	Ignore the error
1	Reset retry - fatal
2	End of block - terminate the element
3	Fatal error - terminate the list
4	unused
5	Reset retry - ignore
6	Busy retry - fatal
7	Busy retry - ignore

Error response codes 4 and 8 through 15 currently perform the same function as ERC 3. The user should avoid using these codes since they may be defined in the future.

7.3.1 ERC 0 - Ignore The Error

This is a very usefull response for most types of errors. Expected slave status errors could be ignored. Many applications (i.e. physics data acquisition) also like to ignore parity errors. It is not advised that this ERC be used in the response timeout field. If used in the RTO field, chances are the list will terminate with an illegal operation error (SEEIOP). All errors which are ignored are recorded in status element and the warning flag (SEEWRN) is set.

7.3.2 ERC 1 - Reset Retry, Fatal

The FSD can automatically handle retries for all FASTBUS errors. The number of times the FSD will attempt the retry is programmed through the retry counter in the system parameter block. The FSD will do retries in a different mannor depending on the type of FASTBUS device and also on what FASTBUS cycle the error occurs. For primary and secondary addressing errors, the FSD simply repeats the cycle with the same data. Single word data transfers also simply repeat the cycle. Block transfers, however, need to be handled specially since the slave automatically increments the next transfer address (NTA) after each data transfer.

If the device is a FIFO (OPCFIO is set in LSTOPT), the FSD retries the data transfer by shifting to single word mode (mode select = 0). If the data is accepted, the FSD shifts back to block transfer mode (mode select = 1) and transfers the remaining data words.

For all other devices, the FSD detaches (lowers AS), then readdresses the device. If no secondary address cycle was ever issued, the FSD assumes the primary address is a logical address. The total number of words transferred prior to the error are added to the primary address. This modified logical address contains the correct internal address to start transferring at the word in error. If a secondary address cycle had been issued, a primary address cycle with the orignal address is issued. This is followed by a secondary address cycle whose address is the original secondary address plus the number of words transferred correctly. In either case the slave should have an NTA which points at the word in error.

The FSD retries until either the cycle completes without an error, or the maximum retry count is reached. When the FSD runs out of retry attempts, the error is treated as a fatal error (see ERC 3, fatal errors). The status element indicates that a retry was attempted (SEERTY, SEEWRN are set in STEERW).

7.3.3 ERC 2 - End Of Block, Terminate The Element

Errors programmed with this response code cause the FSD to stop executing the current list element. The FSD terminates the element as if no error occurred and proceeds with the next element in sequence. This type of response is useful when doing indeterminant length block transfers. Slaves usually respond with SS=2 when they have no more data. This code would terminate the element as if the user had asked to transfer only the total number of words of data available from the device. The user could also use this code for primary address response timeouts to skip a list element whose primary address does not correspoind to any FASTBUS device.

7.3.4 ERC 3 - Fatal Error, Terminate The List

This code will cause the FSD to abort any further execution. If the FASTBUS Operation was a write, the UNIBUS buffer pointer or the FASTBUS transfer device address are set to point at the word in error. If the FASTBUS Operation was a read, all data is written to the UNIBUS buffer or FASTBUS transfer device. This includes the word in error, however the buffer pointer is set to point at the word in error.

7.3.5 ERC 5 - Reset Retry, Ignore

This code performs a retry in the same mannor as ERC 1. The difference however is the way the FSD treats errors which cannot be corrected by retrying. If the FSD runs out of retry attempts the error is ignored in the same fashion as ERC 0.

7.3.6 ERC 6 - Busy Retry, Fatal

Busy retries are handled the same as reset retries except for block transfers. If the error occurred during a block transfer cycle, no reset of the device is performed; the cycle is simply repeated. This error code is usefull for errors such as SS=1 where the NTA is known not to have incremented. As with ERC 1, the list element aborts with a fatal error if the error cannot be corrected.

7.3.7 ERC 7 - Busy Retry, Ignore

This error reponse code performs the same action as ERC 6 except that uncorrectable errors are ignored.

CHAPTER 8

MULTIPLE DEVICE ADDRESSING

Many times in data acquisition and control, the same function needs to be performed on a large number of devices. The user could generate a list containing a separate list element for each device which needs to be addressed. However there is a much easier way to do this with the FSD. The FSD has a feature called Mutiple Device Addressing (MDA). This feature allows the user to perform the same standard opcode on a list of devices which had previously been programmed into the FSD. The list of devices is loaded from the UNIBUS buffer into FSD internal memory. Since the FSD does not have to fetch a new list element from the UNIBUS for each device, total throughput is increased.

8.1 USING MULTIPLE DEVICE ADDRESSING

Prior to executing any MDA opcodes, the list of addresses must be loaded into the FSD. The list is loaded via special function opcode 014. The list is stored in memory internal to the FSD. The list can contain between 0 and 63 different device addresses. These addresses are used during the primary address cycle of the opcode which utilizes the MDA table. If no addresses are given, executing an MDA opcode will result in a null operation. Initially the table is empty.

Once the table has been loaded the user can execute any MDA opcode. The opcode is executed once for each address contained within the table. These addresses are used during the primary address cycle. The primary address contained within the list element (LSTPAL/LSTPAH) is ignored. Secondary addresses are still taken from the list element (LSTSAL/LSTSAH). Block transfer word counts and immediate data are also taken from the list element (LSTWCL/LSTWCH).

All option bits in LSTOPT and LSTOPC are honored. These include OPCHFB, OPCHFA and OPCHFX. Normally the FSD will release FASTBUS between execution of the opcode on the different addresses in the table. If however any of the hold bits are set, the bus and/or device will not be released. Block transfer burst rules are also obeyed by the FSD.

8.2 MDA BUFFER MANAGEMENT

MDA opcodes use the buffer in the same manner as if all the addresses in the MDA table had their own list element. For read opcodes, all data read from the FASTBUS devices is packed in the buffer in sequential order. For write opcodes, the buffer pointer is moved as data is written to the FASTBUS device. Because of the way write operations are done, indeterminate block transfer writes with an MDA opcode could have unexpected results.

8.3 MDA STATUS ELEMENTS

The FSD will output a status element for each address an MDA opcode used during execution. Since this could cause confusion when examining the status block, the FSD also outputs a status element to indicate an MDA opcode was being executed. This MDA header element is output prior to executing the opcode with any address. The information bit SEEMDA is set indicating this is an MDA header element. The status element word contains the number of device addresses stored in the table. This may be the number of status elements which follow in the block which the MDA opcode will output. Due to fatal errors, this will not necessarily be true.

Following the MDA header element, the FSD will write status elements for each primary address the opcode used during execution. The format of the status element is the same as for other standard functions opcodes.

8.4 ERROR HANDLING

MDA opcodes obey the same rules for error handling which other standard function opcodes use. However some clarification is needed for using the end of block and fatal error response codes.

8.4.1 End Of Block Response Code

If the FSD response to an error is the end of block response, the current list element is not aborted as is the case for other opcodes. The FSD terminates the opcode for the current address, writing the status element to the status block. The FSD then goes to the next address in the table instead of the next list element in the list.

8.4.2 Fatal Error Response Code

When a error occurs whose response code is fatal, the FSD aborts the list at the given point. No other addresses within the MDA table are referenced.

CHAPTER 9

FASTBUS TRANSFER ADDRESSING

Many applications of FASTBUS require that data be moved from one FASTBUS device to another. It is also desirable that this motion of data be done at FASTBUS transfer rates. However moving the data into the UNIBUS buffer through the FSD forces the data to be transferred at UNIBUS rates. For this reason the FSD has a feature called FASTBUS Transfer Addressing (FTA). This allows the FSD to move data between two different FASTBUS devices without moving the data into the UNIBUS buffer. The second FASTBUS device performs the same function as the UNIBUS buffer.

9.1 FASTBUS TRANSFER ADDRESSING DEVICE

Almost any FASTBUS device can be used as the buffer in FASTBUS Transfer Addressing opcodes. There are only two restrictions placed on the FASTBUS Transfer Device.

1. The addressing space within the FASTBUS Transfer Device used for the buffer must be in FASTBUS data space. Transfers to control space are not supported.

2. The buffer must be accessable by direct addressing. Indirect addressing modes, such as is used to access the Segment Interconnect Route Table, are not supported.

To summarize, the FASTBUS Transfer Device must be similar to a random access memory module. However first in, first out type of devices can be used as the buffer.

9.2 USING FASTBUS TRANSFER ADDRESSING

Prior to executing any opcodes which use the FASTBUS Transfer Addressing feature, the user must define the FASTBUS Transfer Device (i.e. the buffer). This is done via special function opcode 060. This list element tells the FSD the primary address, secondary address and size of the FASTBUS Transfer Device. The option bits OPCFIO and OPCFTS are also used by the FSD to determine certain properties of the device. If the device is a FIFO, OPCFIO must be set to correctly handle error retries during block transfers. If a secondary address must be issued prior to any data transfers (i.e. the primary address is a geographical address), then OPCFTS must be set. All other options bit are ignored.

Once the FASTBUS Transfer Device has been defined, the user can execute any standard function opcode which uses the FTA. It is also possible to use special function opcodes which do primitive FASTBUS data transfers with the FASTBUS Transfer Device.

9.3 FASTBUS TRANSFER ADDRESSING BUFFER MANAGEMENT

Once defined, the FSD will remember the FASTBUS Transfer Device between lists. This includes the number of words transferred. As with the UNIBUS buffer, data is always sequentially read and written to the device. The user can read the current definition of the FASTBUS Transfer Device as well as the number of words read/written to it since the last definition. This can be done via special function opcode 061.

9.4 FASTBUS TRANSFER ADDRESSING ERROR HANDLING

In the coarse of operations, errors can result during data transmission to the FTD. When this occurs, the FSD uses the same error handling techniques as for other FASTBUS data transfer errors. The same error response words are used and the error response codes have the same meaning. Since the FSD buffers data internally when transfering the data, it is possible that data can be lost if fatal or end of block errors occur when transfering to the FASTBUS Transfer Device.

APPENDIX A

ACKNOWLEDGEMENTS

Design for the FASTBUS Segment Driver microcode was prepared with help of the following people.

R. Brown, R. Downing, M. Haney
B. Jackson, K. Nater and J. Wray
University of Illinois
Loomis Labratory of Physics
1110 W. Green Street
Urbana, Illinois 61801

E. Barsotti, M. Larwill, T. Lagerlund,
R. Pordes and L. Taff
Fermilab, P. O. Box 500
Batavia, Illinois 60510

D. Gustavson, C. Logg
Stanford Linear Accelerator Center
Stanford University
Stanford, California 94305

INDEX