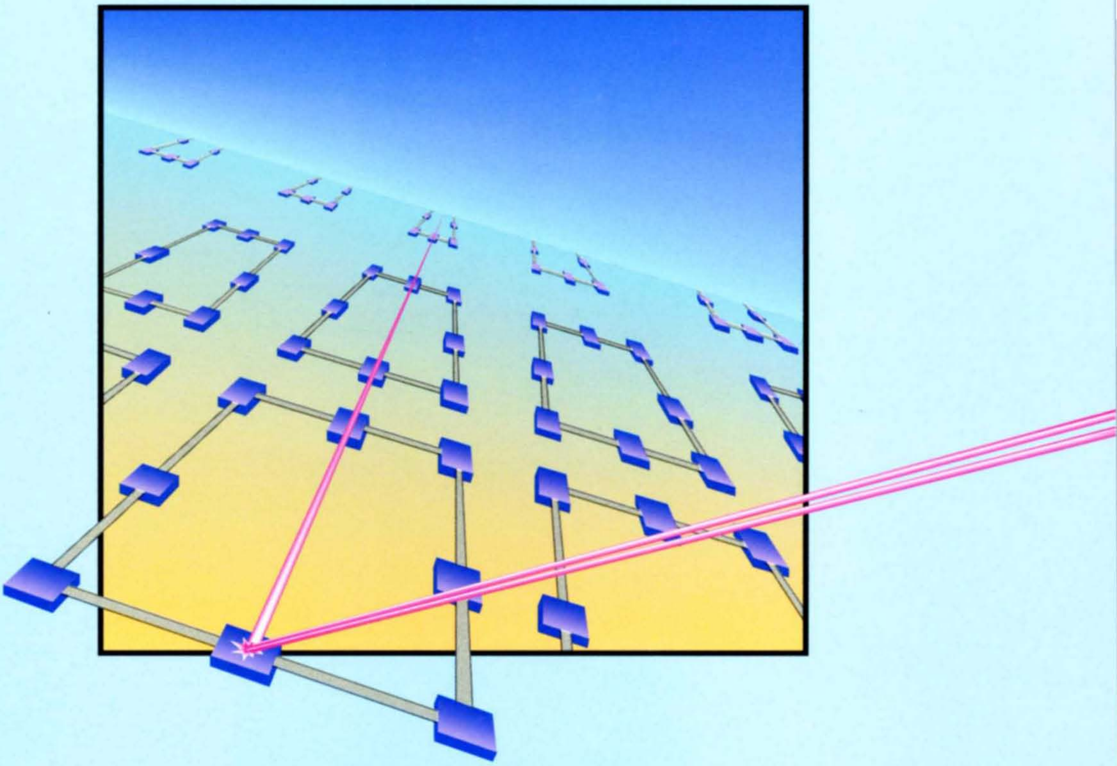
MC68839UM/AD

FDDI



FDDI System Interface User's Manual

MC68839

MOTOROLA

MOTOROLA

MC68839 FSI

User's Manual

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and Ⓜ are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

TABLE OF CONTENTS

Paragraph Number | Title | Page Number

Section 1
Introduction

Section 2
Functional Block Description

Section 3
Registers

TABLE OF CONTENTS (Continued)

Section 4
Signal Descriptions

Section 5
Commands and Indications

TABLE OF CONTENTS (Continued)

Section 6
Functional Operation

TABLE OF CONTENTS (Continued)

Section 7
Initialization, Programming and Examples

Section 8
Port Operation

TABLE OF CONTENTS (Continued)

Paragraph Number | Title | Page Number

Section 9
Boundary Scan

Section 10
Electrical Specifications

Section 11
Ordering Information and Mechanical Data

TABLE OF CONTENTS (Concluded)

Paragraph Number	Title	Page Number

LIST OF ILLUSTRATIONS

Figure Number | Title | Page Number

LIST OF ILLUSTRATIONS (Concluded)

LIST OF TABLES

SECTION 1
INTRODUCTION

Motorola's FDDI chipset consists of an FDDI Clock Generator (FCG) timing and data recovery circuit, an Elasticity and Link Management (ELM) physical layer circuit, a Media Access Control (MAC) circuit, and an FDDI System Interface (FSI). The relationship of the circuits in this chipset is detailed in Figure 1.

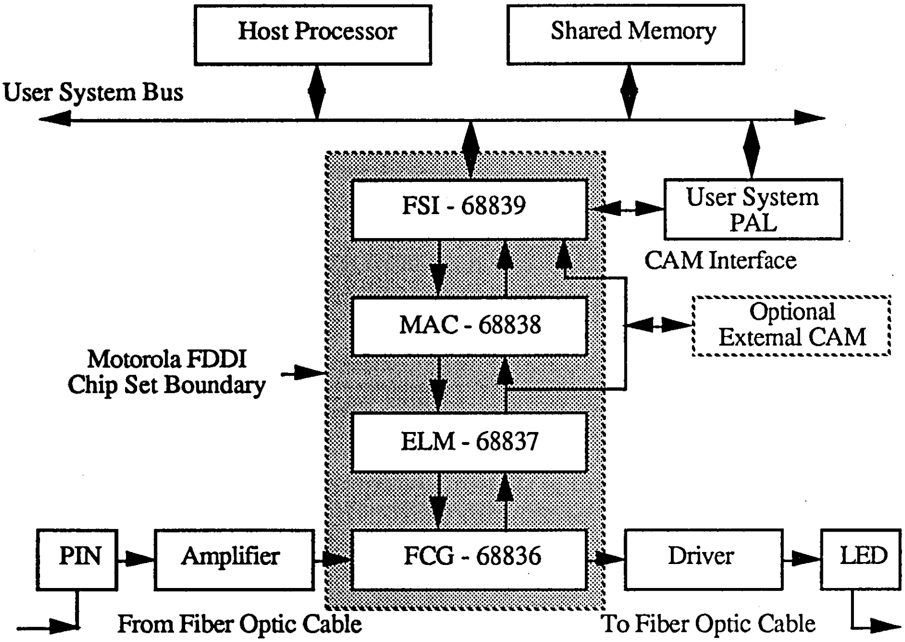


Figure 1-1. Motorola FDDI Chip Set

This document details the FDDI System Interface (FSI) functionality. Features of the FSI include:

- Supports up to a 50 MHz two-clock processor access cycle time or 25 MHz system bus access cycle time (40 nsec cycle time)
- Can easily connect to any bus including high speed processors, little- and big-Endian busses, and multiplexed / non-multiplexed address / data busses

- Programmable partitioned 8K byte internal RAM for temporary data storage (up to 1K of this RAM is used for internal FSI management)
- Four fixed priority transmit rings support synchronous and asynchronous frames
- Receive filtering by Frame Control field into two different rings
- Logically separate, programmable, 32-entry, 48-bit internal CAM handles Destination or Source, Individual or Multicast addresses
- Ring memory structure with multiple buffers per frame
- Data Path parity generation and checking
- Two identical 32-bit ports allowing multiple configurations including 64-bit operations

The FDDI System Interface was designed such that external, FDDI-specific memory is not required. There are 8K bytes of on-chip memory. The user decides how much of it is dedicated to transmission and how much is dedicated to reception. Due to the large amount of on-chip memory, user system bus latencies of approximately 80 microseconds may be supported with all receive and transmit FIFOs active, or approximately 250 microseconds with a single transmit and single receive FIFO active. (More details on the ability of the FSI to support high latency systems can be found in Appendix C.) Local memory may be added to support even longer system latencies.

In order to be able to connect to any type of microprocessor bus, bus control logic is provided by the user off chip, typically with a programmable array logic (PAL) device. The required logic is straightforward.

Two programmable 32-bit ports are provided to support a variety of possible configurations. As examples:

a. one port may be used for SMT frames and the second for other frames including data frames, or
b. both ports may be combined into a single 32- or 64-bit data port to interface to a special host processor, or
c. one port may be used for access only to buffer descriptors stored in a small memory while the second port may be interfaced to a common memory used to buffer the data.

For further examples of the different configurations that may be supported as well as more detailed explanations of the above configurations, refer to Appendix B.

SECTION 2
FUNCTIONAL BLOCK DESCRIPTION

The FDDI System Interface has four main blocks: the MAC Interface Unit, the Internal Memory Array, the Port Control Unit, and the Main Control Unit.

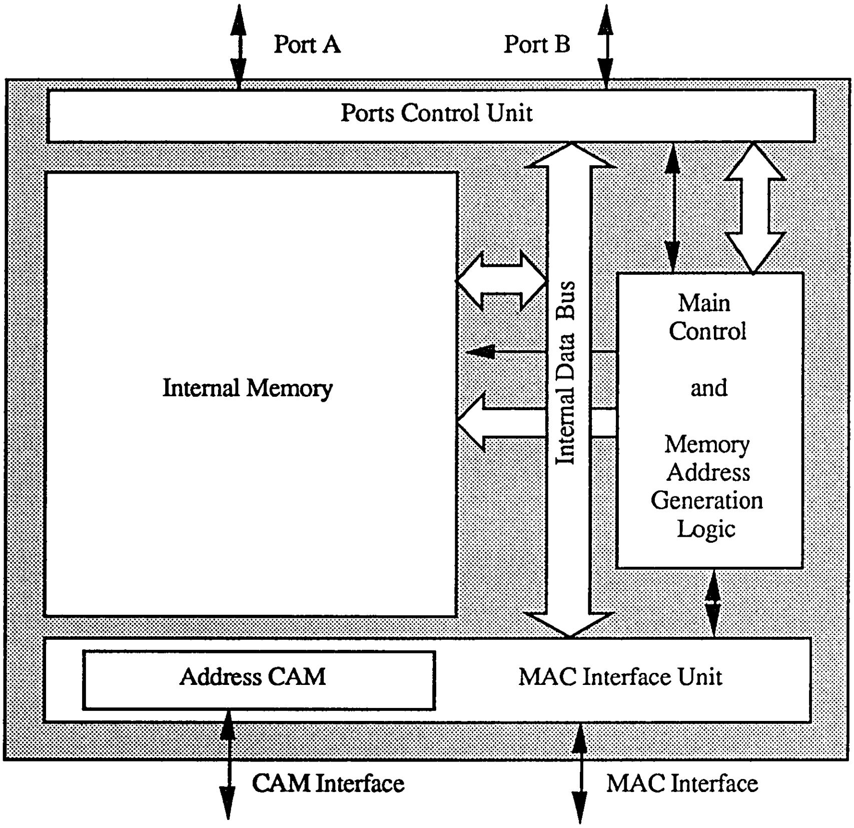


Figure 2-1. Internal Architecture

2.1 MAIN CONTROL

The Main Control Unit is responsible for controlling the other blocks and the timing between them. It performs processor command decoding and execution, internal virtual memory address generation, and handles priorities between various memory array accesses.

2.2 MAC INTERFACE

The MAC interface is responsible for interfacing the FSI to the MAC chip and transfers data between the internal memory and the MAC chip. This block generates and receives all the signals necessary for these transfers. The MAC interface contains separate receive and transmit busses, each of which can transmit either data or control information. Transfers on these busses are synchronous with BYTCLK.

The MAC interface also contains small hardware FIFOs for transmission and reception. During normal FSI operation, frames which are not meant for this station are discarded during an early stage of processing while still inside the receive hardware FIFO in the MAC interface. When receiving data, the MAC interface transfers the information from the hardware FIFO in blocks of 8 bytes into the internal memory array. During data transmission, the MAC interface gets the control information and frames from the internal memory in blocks of 8 bytes which are transferred to the hardware transmit FIFO in the MAC interface.

The MAC interface includes a selector mechanism that operates on the received frames as follows: when the MAC interface starts to receive data, it decodes the Frame Control (FC) field of the frame being received and, according to the value of the FC, determines into which receive ring to transfer the data. This receive ring selection is programmable by the user upon initialization.

The MAC Interface also has three rejection signals: receive reject (input), receive abort (output) and transmit abort (input). The rejection signals may be used when the MAC is configured in promiscuous mode, since it enables external logic to decide whether or not to receive the frame based on any arbitrary pattern or algorithm implemented by that external logic.

2.2.1 CAM INTERFACE

The CAM Interface unit contains a 32-entry by 48-bit CAM which is used for Address matching. The CAM includes a valid bit with each CAM entry. The CAM Interface control signals include a load address signal to mark the beginning of a received address field and an output indication signal if an address is matched.

The CAM has two access paths: the main control port, which includes address and data for setting up and controlling CAM entries, and the MAC interface, which takes data from the PHDAT pins in the ELM to be compared with the contents of the CAM.

The CAM is manipulated through the CAM commands. As a result of these commands, the system or node processor can write, read or clear an entry in the CAM or compare data with the CAM contents. A write operation of a CAM command updates the contents of the entry pointed to by the command with the data supplied in the command and either sets the corresponding valid bit to indicate a valid address or resets the valid bit to clear or invalidate the entry. A read operation of a CAM command supplies the Host processor with the contents of the entry pointed to by the command.

2.3 INTERNAL MEMORY

The Internal Memory is a memory array of 8K bytes arranged as 512 rows of 128 bits plus parity. It is used to temporarily store transmit and receive frames and descriptors. All the data FIFOs supported by the FDDI System Interface are mapped inside this memory array. The memory array is shared between Port A, Port B and the MAC interface. Memory address generation is centralized inside the main controller.

2.3.1 Internal Transmit Data Structures and Command Issuance

For each Descriptor Ring in external memory, the FSI has two internal FIFOs in the FSI internal memory. One is an Internal Data FIFO for temporary data storage, the other is an Internal Ring FIFO for temporary storage of commands, buffer descriptors and indications.

Commands can also be issued directly to the FSI through the command register as opposed to through the Descriptor Rings as described above. When transmit commands are issued directly to the Command Register, an Internal Command Data FIFO is used for temporary storage of transmit data.

The Internal Data and Internal Ring FIFOs exist only if their rings are defined. When a Transmit Ring is complete (there are no more frames to transmit), its internal Data and Ring FIFOs occupy 64 bytes (32 bytes each) of internal memory. During operation, the size of a Transmit FIFO changes dynamically. If the Transmit FIFO is not transmitting, the maximum allowed Transmit Data FIFO length is defined by its "watermark." The maximum Transmit Ring FIFO length is 160 bytes (5 blocks of 32 bytes). When the Transmit Data FIFO is transmitting, the maximum size of that FIFO inside the internal memory is twice its watermark and the maximum Transmit Ring FIFO length can be 288 bytes (9 blocks of 32 bytes). Note that only one Transmit FIFO will ever be transmitting data at any time.

2.3.2 Internal Receive Data Structures

Much like the transmit side, the FSI has two internal FIFOs in the FSI internal memory for each Receive Ring in external memory. One is an Internal Data FIFO for temporary data storage of received data, and another is an Internal Ring FIFO for temporary storage of Receive Descriptors and Indications. Once the Receive Ring is defined and the receive operation is enabled for a Ring, the data received from the network is saved inside the Internal Data FIFO. If the Receive Ring is Ready, the FSI reads Buffer Descriptors and transfers the data into the location pointed to by the buffer descriptor. The internal memory space reserved for each receive Ring is used for both the Internal Data FIFO and the Internal Ring FIFO.

In addition to received frames, other receiver event indications (e.g., Token Cycle End and My Frame Indications) may occupy the Internal Ring FIFO using one entry (8 bytes) per indication.

2.4 PORT CONTROL UNIT

The Port Control Unit is responsible for data transfers between the internal memory and the host processor. Each port has its own external address generator and a block counter to support DMA operation. (Note that external bus timing is provided by the bus control logic outside the FDDI System Interface.)

2.4.1 Port Operation

Each port of the FSI Port Control Unit may handle both input and output transfers at the same time. The FSI implements an internal priority between the ports in order to schedule the next required transfer request. When the FSI is ready to execute a transfer, it indicates the transfer type by asserting the Request Signals for that port. Once the Request Signals are set, they will not change until the access cycle is complete. A change of Request Signals from an active request to either some other request, or to the Idle state may be done only during the data read/write. At the end of the data cycle, the Chip Select signal is negated and the next cycle request may be sampled by the Bus Control Logic. Note that only the accesses to the Port Data Register have an effect on the port state and on address updating. All other registers may be accessed at any time regardless of the condition of the Request Signals.

Data is transferred through Ports A and B as block transfers. On transmission, when a block of data is transferred into the internal memory, the Port Control Logic assembles the data into 16-byte blocks with internal byte parity and then performs byte-swapping dependent upon whether the bus connected to the port is Big-Endian or Little-Endian. The FSI fully supports byte-swapping, even for 64-bit data. Since the byte-swapping capability is selectable per port, it is possible to use processors with two different byte orders within a single system. After the 16 bytes of information are assembled, they are transferred to the appropriate location in the internal memory. When the frame to be transmitted includes many buffers, the Port Control Logic is responsible for assembling all the data of the frame sequentially such that the data is seen in the internal memory as a contiguous block.

When the block transfer is from the internal memory to the system (reception), the Port Control Logic is responsible for synchronizing between the internal and external transfers. The data is transferred from the internal memory to the port in blocks of 16 bytes.

The accesses to the Ring are handled by the Port Control Unit which generates the address for each one of the rings in order to read commands and descriptors, and to write indications.

The address counter of the external memory address is updated each time the data is read or written by either an external processor or the bus control logic. The address itself may be accessed by selecting the appropriate control pins (see Section 3 on Programming Model).

The ports can be configured to operate separately, 32 bits each, or combined, 64 bits. When configured separately, each of the two ports can be accessed to supply 32-bit data or 32-bit address or 32 bit address and data multiplexed.

The Ports can be configured to operate together in that the address generator of Port A may be accessed from Port A or Port B and vice versa. This allows the Ports to be configured as a single 64-bit data port with 32-bit address multiplexed or as 32-bit data port with 32-bit address not multiplexed.

The Port Control Unit also handles the byte order of the external bus (Little Endian/Big Endian). According to its initial definition, each port is able to transfer the data between the system and internal memory according to the type of bus the FSI is connected to. (See Figure A1 for an example connection to a Motorola 68020 bus.) It must be noted that the command and indication definitions (bit and field assignment), and the addresses in the Ring Buffer Descriptors, are independent of the type of bus the FSI is connected to. Byte order does not matter since the bits and fields are defined explicitly for addresses, commands and indications.

The Port Control Unit can determine whether the addresses of the current location to be accessed in external memory and the previous location accessed in external memory are on the same page. The limits of the page definition are supplied to the FSI on initialization. The result of this comparison is supplied to the external bus controller on the Request Signals so that it can generate the appropriate timing for signals that access a dynamic memory using page or nibble modes.

SECTION 3
REGISTERS

The MC68839 FDDI System Interface contains several registers which may be accessed by the host processor and the Bus Control Logic through both the Port A and Port B data buses. The selection, and programming and/or reading of the registers is accomplished through either port.

The FSI register set consists of two types: a directly addressable set and an internal, indirectly addressable set. The latter set is accessed through the FSI Control Register (FCR).

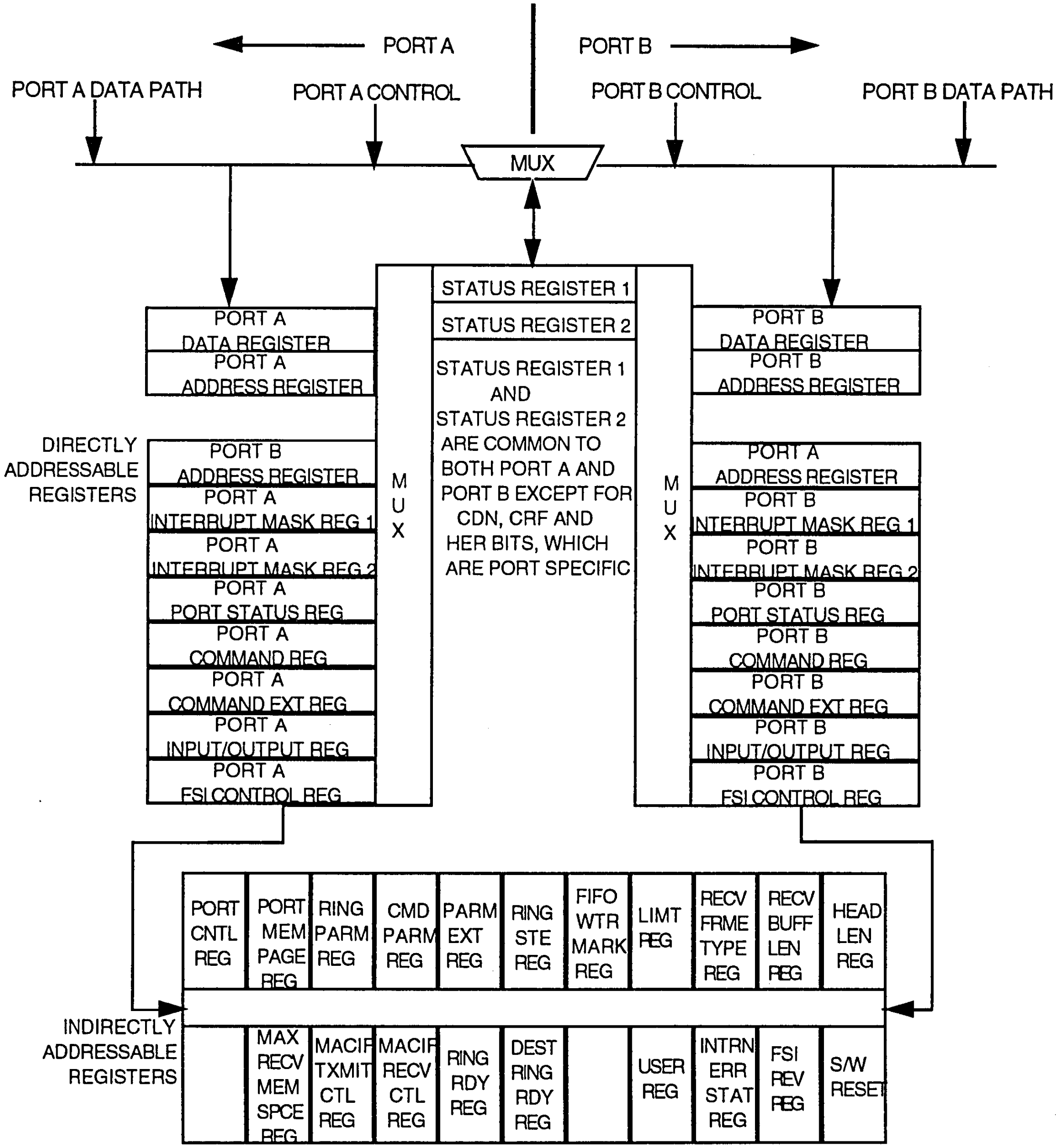


Figure 3-1. Internal and Common Registers

3.1 REGISTER ADDRESS SPACE

Figure 3-2 (below) illustrates the register address space of the FSI. All registers are accessed by means of the CNTL(3:0) signal lines. In addition to host accessible registers, which are discussed in this chapter, the CNTL lines are also used to perform bus control operations (i.e., NOP, ABORT) and data transfer operations as discussed in chapter 7.

Table 3-1. Register Addresses

CNTLx	Read		Write	
3210	Port A	Port B	Port A	Port B
0000	NOP			
0001	Status Register 1			
0010	Data Register	Data Register	Data Register	Data Register
0011	Input/Output Register	Input/Output Register	Input/Output Register	Input/Output Register
0100	Address Register	Address Register	Reserved	Reserved
0101	Interrupt Mask Register 1	Interrupt Mask Register 1	Interrupt Mask Register 1	Interrupt Mask Register 1
0110	Reserved	Reserved	Reserved	Reserved
0111	Port Status Register	Port Status Register	Reserved	Reserved
1000	Address Register (B)	Address Register (A)	Reserved	Reserved
1001	Command Register	Command Register	Command Register	Command Register
1010	Reserved	Reserved	Reserved	Reserved
1011	Command Extension Register	Command Extension Register	Command Extension Register	Command Extension Register
1100	Interrupt Mask Register 2	Interrupt Mask Register 2	Interrupt Mask Register 2	Interrupt Mask Register 2
1101	Status Register 2			
1110	ABORT			
1111	FSI Control Register	FSI Control Register	FSI Control Register	FSI Control Register

3.2 DIRECTLY ACCESSED REGISTERS

3.2.1 Data Register (DTR) (xCNTL=0010)

Each port has its own 32-bit data register used to transfer either data for transmit and receive frames; or commands, descriptors and indications from/to Transmit and Receive Rings.

The allowable access type (Read/Write) to the Data Register depends upon the current Port State and the operation desired. If an FSI Port is in the Read state, the system needs to furnish Transmit Data or Transmit Commands and Descriptors and is only allowed to perform a write access. Note: In all descriptions, port states (Read or Write) are relative to the FSI. Conversely, if an FSI Port is in the Write state, the system is expected to process Receive Data or Indications and is only allowed to perform a read access. Any attempt to access the Data Register in a different manner may result in conflicting behavior of the FSI.

The Data Register is 32-bit. When the Port's Data Registers are concatenated in 64-bit operation mode, the Port A Data Register holds the most significant portion of the data and the Port B Data Register holds the least significant portion of the data. The Data Register access will update the address of the corresponding Port. In 64-bit mode the address is generated in Port A but can be accessed through either port.

3.2.2 This Port's Address Register (ADRx) (xCNTL=0100)

Each Port has an Address Register which is used by the Port Control Unit to access external memory and contains the contents of that port's Address Generator.

The Address Generator has two modes of operation, depending on the nature of the access: Descriptor Ring access or Data access. During a Descriptor Ring access, the Ring Read Address or the Ring Write Address are used as defined in the Define Ring Command. The Ring Address is then incremented, as required, up to the user-defined Ring Maximum Length (RML) value before wrapping at the specified limit.

During a Data access, the address is generated from the Address field of a Buffer Descriptor and is incremented after every Data access. When the transfers are to or from Local Memory the generated address will wrap at the boundary of the local memory space. Accesses to the Address Generator have no influence on the address itself. Only a successful Data access causes a change in the address. The address is advanced according to the Data access width (32- or 64-bits). The address will not be updated in the following situations:

a. When reading the entries of the Ring, the Ring Read Address will not be incremented if the read entry does not belong to the FSI, (i.e., the Ring became Empty).

b. When reading the entries of the Ring, the Ring Read Address will not be incremented if a Parity Error or Port Operation Error are recognized. The address will continue to point to the descriptor which caused the Error.

c. When a First or Last Bit error situation is encountered, see section 7.3.2.

The address register contents have no meaning when the External Request Lines are in the IDLE state.

3.2.3 The Other Port's Address Register (ADRX). (xCNTL=1000)

The Address Generator of one Port may be accessed through the other Port to allow simultaneous access of address and data, (i.e., as in the case of a non-multiplexed 32-bit address and data external bus).

3.2.4 Status Register 1(SR1) (xCNTL=0001)

This register includes all the FSI general status information which may cause an interrupt if the appropriate Interrupt Mask bit is set. There is only one SR1 in the FSI. Access to this register may be performed from either Port A or Port B. The CDN, CRF, and HER bits indicate the status of the port which is reading the SR1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CDN	CRF	HER	IOE	ROV5	ROV4	POEB	POEA	0		RER5	RER4	RER3	RER2	RER1	RER0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		RXC5	RXC4	RCC3	RCC2	RCC1	RCC0	0		RNR5	RNR4	RNR3	RNR2	RNR1	RNR0

The bits in this register are divided between real status bits, which are set and cleared by the FSI according to the status of the item, and sticky bits, which may only be set by the FSI when the appropriate condition(s) occur(s) and are cleared by a host processor write access to the SR1 with the corresponding bit set.

The numbers in parentheses in the following bit descriptions correspond to the appropriate ring number: Receive Rings (5 and 4) and Transmit Rings (3, 2, 1, and 0).

CDN—Command Done

When the SR1 is read from one of the ports, the CDN bit references the Command Register of this port. When a command is written to the Command Register, its Own bit should be set which causes the reset of the CDN bit in the SR1. When command execution has been completed, the FSI writes the indication to the command register with the Own bit reset. This will cause the CDN to be set. CDN is a real time status bit and is not affected by writing to the SR1.

CRF—Control Register Free

When the SR1 is read from one of the ports, the CRF bit indicates the status of the respective port FCR (FSI Control Register). When the CRF bit is set, the FCR of the port may be written. Note that there are many internal control registers which may be set through a Control Register access. These Internal Control Registers can be accessed through either port. The CRF bit is a real time status bit and is not affected by writing to the SR1.

HER—Host Error

The HER bit is set when a parity error is detected (if the parity checking is enabled) during a write access to a directly accessible register. This bit is a sticky bit and must be cleared by the host..

IOE—Internal Operation Error

The IOE bit is set when an internal operation error is detected. This bit is a sticky bit and must be cleared by the host.

Internal Errors may be detected in the following situations:

a. The MAC Interface recognizes a parity error on transmit data. An internal parity error detected during data reads from the internal memory will always indicate a problem in an internal FSI data path. In this case the Transmit Parity Error (TPE) bit is set in the Internal Error Status Register (IER), the frame currently being transmitted is aborted with an appropriate indication and the Transmit Enable (TE) bit in the MACIF Transmit Control Register (MTR) is reset.

b. The Port Interface recognizes a parity error on received data being transferred to the System Bus. The Port Internal Error (PAE or PBE) bit is set in the Internal Error Register.

c. The MAC Interface experiences an internal underrun or an internal overrun. During normal operation, the Main Controller assures a maximum response time for data transfers between the MAC Interface and the internal memory. If this response time exceeds these limits, this might indicate that the Main Controller is not performing properly. As a result, the Internal Overrun Error (IOE) or the Internal Underrun (IUE) bit is set in the Internal Error Status Register (IER). If it is an internal underrun, the currently transmitting frame is aborted with the appropriate indication and the transmitter is disabled (TE reset). If it is an internal overrun error, the receiver is disabled (both RE4 and RE5 reset) and the RABORT signal is generated.

d. The MAC Interface receiver recognizes a protocol handshake error between the MAC and the FSI. In this case, the MAC Error (MER) bit is set in the Internal Error Status Register (IER). The MACIF will generate an RABORT to the MAC and will try to resynchronize with the MAC on the next frame. This error may be caused by some temporary problems (e.g., noise on control lines), therefore no special recovery functions are required. If there are permanent problems in the FSI to MAC interface (e.g., control lines are disconnected or forced to some constant value), the Host processor should perform an external diagnostic.

e. The FSI has experienced an internal memory overrun causing the Memory Overrun (MOV) bit in IER to be set. This error may be caused by an incorrect definition of an internal FIFOs' Watermark. Therefore, the FSI parameter definitions should be checked and the FSI should be reinitialized.

ROV(5-4)—Receive Overrun Error

There is one ROV bit for each Receive Ring. Each bit is set when an overrun condition occurs for its Ring. These bits are sticky and may be cleared only by a host write to the SR1.

POEB—Port Operation Error B

The POEB bit is set when an operation error is recognized during a Port B operation. The POEB bit is a sticky bit and must be cleared by the host.

POEA—Port Operation Error A

The POEA bit is set when an operation error is recognized during a Port A operation. The POEA bit is a sticky bit and must be cleared by the host..

RER(5-0)—Ring Error

The RER bit is set if a Parity error, First or Last Bit error or a Port Operation error has been recognized during an access to the entry of a Ring. These bits are sticky bits and must be cleared by the host..

RXC(5-4)—Receive Complete

Each RXC bit refers to one of the Receive Rings. The RXC bit is set when a buffer indication is written with the Last bit set to the Ring; i.e., the receive frame has been completely transferred to the external memory and passed to the system processor. These bits are sticky bits and must be cleared by the host.

RCC(3-0)—Ring Command Complete

Each RCC bit is referenced to one of the Transmit Rings. The RCC bit is set in two cases:

a. The indication of a transmit frame with the Last bit set has been written back into the ring (i.e., a frame has been completely transmitted).
b. A command from a Transmit Ring has been executed and its indication has been written back into the Ring. These bits must be cleared by the host.

RNR(5-0—Ring Not Ready

Each RNR bit is set when any condition in the FSI causes the ring to become Not Ready, even if the ring is currently Not Ready. These bits are sticky bits and must be cleared by the host..

3.2.5 Status Register 2 (SR2) (xCNTL=1101)

This register includes the FSI status information which may cause an interrupt if the appropriate Interrupt Mask bit is set. There is only one SR2 in the FSI. Access to this register may be accessed from either Port A or Port B.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0	DRE7	DRE6	0	0	DRE3	DRE2	DRE1	DRE0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DXC7	DXC6	0	0	DXC3	DXC2	DXC1	DXC0	DNR7	DNR6	0	0	DNR3	DNR2	DNR1	DNR0

All the bits are sticky bits, which may only be set by the FSI when the appropriate condition(s) occur(s) and are cleared by a host processor write access to the SR2 with the corresponding bit set.

INT(7-0—User Defined Interrupt

Each one of these status bits may be set by writing to SIG internal control register. These interrupts may be used for signaling between Host and Node processors.

DRE(7,6,3-0)—Destination Ring Error.

The DRE bit is set if a Parity or Port Operation Error has been recognized during an access to the entry of a Destination Ring.

DXC(7,6,3-0—Destination Transfer Complete

Each DXC bit is referenced to one of the Destination Rings. The DXC bit is set when the "Last" indication (with the Last bit set) has been written back into the Destination ring (i.e., a frame has been completely transferred).

DNR(7,6,3-0)—Destination Ring Not Ready

Each DNR bit is set when its Destination Ring has changed its state from Ready to Empty state.

3.2.6 Interrupt Mask Register 1(IMR1) (xCNTL=0101)

There are two IMR1s, one for each Port. There is an Interrupt Enable bit for each status bit in the SR1. The FSI will furnish an interrupt when both the SR1 bit and its respective Interrupt Enable bit in that Port's IMR1 are set. If the same Interrupt Enable bit is set in both IMR1s, the interrupt is generated on both Ports as a result of a SR1 bit becoming set. The IMR1s are cleared on power-up reset and unaffected by a software reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CDE	CFE	HEE	IEE	RVE5	RVE4	PEEB	PEEA	0		REE5	REE4	REE3	REE2	REE1	REE0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		RXE5	RXE4	RCC E3	RCC E2	RCC E1	RCC E0	0		RNE5	RNE4	RNE3	RNE2	RNE1	RNE0

CDE—Command Done Interrupt Enable

When the CDE and CDN are both set, an interrupt is generated.

CFE—Control Register Free Interrupt Enable

When the CFE and CRF are both set, an interrupt is generated.

HEE—Host Error Interrupt Enable

When the HEE and HER are both set, an interrupt is generated.

IEE—Internal Operation Error Interrupt Enable

When the IEE and IOE are both set, an interrupt is generated.

RVE(5-4)—Receive Overrun Error Interrupt Enable

When the RVE(i) and ROV(i) are both set, an interrupt is generated.

PEEB—Port Operation Error B Interrupt Enable

When the PEEB and POEB are both set, an interrupt is generated.

PEEA—Port Operation Error A Interrupt Enable

When the PEEA and POEA are both set, an interrupt is generated.

REE(5-0)—Ring Error Interrupt Enable

When the REE(i) and RER(i) are both set, an interrupt is generated.

RXE(5-4)—Receive Complete Interrupt Enable

When the RXE(i) and RXC(i) are both set, an interrupt is generated.

RCCE(5-0)—Ring Command Complete Interrupt Enable

When the RCCE(i) and RCC(i) are both set, an interrupt is generated.

RNE(5-0)—Ring Not Ready Interrupt Enable

When the RNE(i) and RNR(i) are both set, an interrupt is generated.

The IMR1 may be read any time; therefore a read-modify-write operation may be performed by the host processor in order to change the interrupt mask.

3.2.7 Interrupt Mask Register 2(IMR2) (xCNTL=1100)

There are two IMR2s, one for each Port. There is an Interrupt Enable bit for each status bit in the SR2. The FSI will furnish an interrupt when both the SR2 bit and its respective Interrupt Enable bit in that Port's IMR2 are set. If the same Interrupt Enable bit is set in both IMR2s, the interrupt is generated on both Ports as a result of a SR2 bit becoming set. The IMR2s are cleared on power-up reset and unaffected by a software reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INE7	INE6	INE5	INE4	INE3	INE2	INE1	INE0	DREE 7	DREE 6	0	0	DREE 3	DREE 2	DREE 1	DREE 0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DXCE 7	DXCE 6	0	0	DXCE 3	DXCE 2	DXCE 1	DXCE 0	DNR E7	DNR E6	0	0	DNR E3	DNR E2	DNR E1	DNR E0

INE(7-0)—User Defined Interrupt Enable

When the INE(i) and INT(i) are both set, an interrupt is generated.

DREE(7,6,3-0)—Destination Ring Error Interrupt Enable

When the DREE(i) and DRE(i) are both set, an interrupt is generated.

DXCE(7,6,3-0)—Destination Ring Transfer Complete Interrupt Enable

When the DXCE(i) and DXC(i) are both set, an interrupt is generated.

DNRE(7,6,3-0)—Destination Ring Not Ready Interrupt Enable

When the DNRE(i) and DNR(i) are both set, an interrupt is generated.

The IMR2 may be read any time; therefore a read-modify-write operation may be performed by the host processor in order to change the interrupt mask.

3.2.8 Input/Output Register (IOR) (xCNTL=0011)

There are two 32-bit Input Output Registers, one for each Port. These registers may be written and read by the host processor at any time. The IOR has no meaning for the FSI and it is not changed as a result of FSI operation.

The IOR may be used by the host processor to hold an interrupt vector which could be read and used by the host on interrupt detection. Another potential use for the IOR is to extend the address space to 64 bits. On bus accesses the IOR could be read and used as the upper 32 bits of address as desired.

3.2.9 Port Status Register (PSR) (xCNTL=0111)

There are two PSRs, one for each Port. Each PSR indicates the current Port Status and may be read by the host processor at any time. The Port Status Registers are Read Only registers.

31 28	27	26	25	24	23 19	18 16
0	REQ3	REQ2	REQ1	REQ0	0	RNM

15 0
TCN

REQ—Request

These four status bits show the actual state of the Request Output lines of this Port.

REQ (3) differentiates between descriptors or indications (REQ(3) = 0), and data transfers (REQ(3) = 1).

0 = Descriptors or Indications
1 = Data transfers

The encoding of REQ(2:0) is:

000	Idle State
100	Read Cycle
101	Read Cycle with address on the same memory page
010	Write Cycle
011	Write Cycle with address on the same memory page

RNM—Ring Number

RNM indicates the number of the Ring currently being accessed.

000	Transmit Ring 0
001	Transmit Ring 1
010	Transmit Ring 2
011	Transmit Ring 3
100	Receive Ring 4
101	Receive Ring 5
110	Command Register A
111	Command Register B

TCN—Transfer Counter

The TCN has different meanings in the following situations:

a. For a transmit data read operation, the TCN contains the number of bytes remaining to be transferred in the current Transmit Data Buffer.
b. In the case of a receive data write operation, the TCN contains the number of bytes already transferred to the external memory for the frame currently being received.
c. During a Ring FIFO Read operation, the TCN will indicate the number of bytes that the FSI expects to read from the Ring.
d. The TCN has no meaning during a Ring Write operation.

Note that the operation of the Port may be switched between various states according to internal priorities and algorithms. Therefore, in cases where several activities are taking place inside the FSI, the port context may be switched before the transfer counter reaches zero.

3.2.10 Command Register (CMR) (xCNTL=1001)

There are two Command Registers, one for each Port. The CMR is used for direct command access to the FSI, as opposed to a command which is presented in the Transmit Descriptor Ring. The CMR may be written by the host processor any time the CMR is available; i.e., when its CDN bit is set in the Status Register 1, and the required port (as indicated by the CDA bit in the CPR register) is enabled, i.e., PE bit in the corresponding PCR is set to "1". A write access to the CMR when the respective Command Done (CDN) bit is cleared is not allowed. If both CMRs are used for commands, the following cases may occur:

a. If both instructions require the same operation; i.e., two Transmit Frame commands, or two Ring handling commands to the same ring, then these commands are executed sequentially.

b. If the commands can be executed together; i.e., a Transmit Command and CAM Command, etc.; the FSI will execute them concurrently.

The command written to the CMR is normally associated with the Command Extension written to the Command Extension Register (CER), which contains the second long word of the command entry. This second long word is usually the address of a Data Buffer or a Ring. The write access to the CMR will cause the FSI to take this command and to start to execute it. Therefore, the CER should be written prior to the CMR write and should not be changed until the Command Done Indication has been received. Similar to command reads from Rings, direct commands written directly to a CMR have an Own bit which must be written as a one by the host processor. The Own bit is cleared by the FSI when the command has been executed and an indication is written to the CMR and to the Command Extension Register (CER), if applicable, by the FSI. The CMR may be read by the host at any time.

3.2.11 Command Extension Register (CER) (xCNTL=1011)

There are two Command Extension Registers, one for each Port. Each is related to their respective Command Register and contains the second long word, if required, of a command entry. The CER should be changed only when the CDN bit associated with the appropriate Command Register is set, (i.e., the previous command is done), and should be written prior to writing its associated command to the Command Register (CMR). If a parity error occurs (indicated by the HER bit in the port's SR1) during the CER access, the CER should be rewritten prior to writing to the CMR.

3.2.12 FSI Control Register (FCR) (xCNTL=1111)

The FCR is used to access the internal register set of the FSI. The FCR may be written by the host processor at any time when the CRF bit in the SR1 is set.

31	30 27	26 24	23 16
R	IRT	IRI	DATA

15 0
0

When the FCR is written with the R bit (bit 31) zero, its data portion (bits 23-16) is transferred into one of the internal Control Registers according to the Register Identification Fields, (IRT and IRI), after synchronization to the FSI system clock. When data has been transferred, the relevant CRF bit is set again. In order to read data from the internal control registers, the FCR should be written with the R bit set. The FSI system will transfer the value of the relevant internal control register to the FCR, and the appropriate CRF bit is set to indicate that the FCR is ready to be read by the host processor. The format of the FCR is as follows:

R—Read/Write operation
(1 = Read, 0 = Write)

IRT—Internal Control Register Type.

IRI—Internal Control Register ID.

The following is a list of the Internal Control Registers and their associated IRT and IRI number in hexadecimal:

Register Name	IRT	IRI	Number of Registers
Port Control Registers (PCR)	B	6, 7	2
Port Memory Page Registers (PMP)	8	6, 7	2
Ring Parameters Registers (RPR)	6	0 - 5	6
Parameter Extension Registers (PER)	7	0 - 7	8
Command Parameters Registers (CPR)	6	6,7	2
Ring State Registers (RSR)	E	0 - 7	8
FIFO Watermark Registers (FWR)	4	0 - 7	8
Limit Registers (LMR)	5	0 - 5	6
Receive Frame Type Registers (RFR)	1	4, 5	2
Receive Buffer Length Registers (RBR)	3	4, 5	2
Header Length Registers (HLR)	2	4	1
Maximum Receive Memory Space Registers (RMR)	C	4, 5	2
MACIF Transmit Control Register (MTR)	1	0	1
MACIF Receive Control Register (MRR)	1	1	1
Ring Ready (RDY)	0	0 - 5	6
Destination Ring Ready (DRY)	9	0 - 3,6,7	6
Signal Register (SIG)	D	0 - 7	8
User Register (USR)	F	4- 5	2
Revision Register (REV)	F	2	1
Internal Error Status Register (IER)	F	0	1
Software Reset(SWR)	F	7	1

DATA—Internal Control Register Data

Data for a register write and returned data on a register read.

Table 3-2. FSI Indirect Registers

IRT	IRI							
	000	001	010	011	100	101	110	111
0000	Ring Ready Reg 0	Ring Ready Reg 1	Ring Ready Reg 2	Ring Ready Reg 3	Ring Ready Reg 4	Ring Ready Reg 5	—	—
0001	MACIF Transmit Reg	MACIF Receive Reg	—	—	Receive Frame Type Reg 4	Receive Frame Type Reg 5	—	—
0010	—	—	—	—	Header Length Reg	—	—	—
0011	—	—	—	—	Rx Buffer Length Reg 4	Rx Buffer Length Reg 5	—	—
0100	FIFO Watermark Reg 0	FIFO Watermark Reg 1	FIFO Watermark Reg 2	FIFO Watermark Reg 3	FIFO Watermark Reg 4	FIFO Watermark Reg 5	FIFO Watermark Reg 6	FIFO Watermark Reg 7
0101	Limit Reg 0	Limit Reg 1	Limit Reg 2	Limit Reg 3	Limit Reg 4	Limit Reg 5		
0110	Ring Parameter Reg 0	Ring Parameter Reg 1	Ring Parameter Reg 2	Ring Parameter Reg 3	Ring Parameter Reg 4	Ring Parameter Reg 5	Command Parameter Reg A	Command Parameter Reg B
0111	Parameter Extension Reg 0	Parameter Extension Reg 1	Parameter Extension Reg 2	Parameter Extension Reg 3	Parameter Extension Reg 4	Parameter Extension Reg 5	Parameter Extension Reg 6	Parameter Extension Reg 7
1000	—	—	—	—	—	—	Port Memory Page Reg A	Port Memory Page Reg B
1001	Destination Ready Reg 0	Destination Ready Reg 1	Destination Ready Reg 2	Destination Ready Reg 3	—	—	Destination Ready Reg 6	Destination Ready Reg 7
1010	—	—	—	—	—	—	—	—
1011	—	—	—	—	—	—	Port Control Reg A	Port Control Reg B
1100	—	—	—	—	Maximum Rx Memory Space Reg 4	Maximum Rx Memory Space Reg 5	—	—
1101	Signal Reg 0	Signal Reg 1	Signal Reg 2	Signal Reg 3	Signal Reg 4	Signal Reg 5	Signal Reg 6	Signal Reg 7
1110	Ring State Reg 0	Ring State Reg 1	Ring State Reg 2	Ring State Reg 3	Ring State Reg 4	Ring State Reg 5	Ring State Reg 6	Ring State Reg 7
1111	Internal Error Status Reg	—	Revision Reg	—	User Reg 0	User Reg 1	—	Software Reset

3.3 INDIRECTLY ACCESSED REGISTERS

The following subsections detail the indirectly accessed internal control registers which are used to control FSI operation and to define various parameters. These control registers are 8-bits in width and are accessed indirectly through either port's FSI Control Register (FCR). The addresses shown in this section refer to the IRT and IRI values in the respective FCR fields.

3.3.1 Port Control Register (PCR) (IRT=B; IRI=6,7)

There are two PCRs, one for each Port. (IRI = 6 for port A, and IRI = 7 for port B.)

7	6	5	4	3	2	1	0
HPE	PC	SO	DO	0	DW	0	PE

HPE—Host Parity Enable

This bit indicates whether there is parity checking for Port accesses that involve the host processor directly; i.e., SR1,SR2, FCR, CMR, CER, IMR1, IMR2 or IOR write accesses. If this bit is set, parity checking is enabled.

PC—Parity Control

This bit in PCR of Port A defines the parity treatment mode for the entire FSI. If this bit is zero, the parity is odd. If this bit is set, the parity is even.

SO—Synchronous Operation

If this bit is reset, the assertion of the Request Output signals when the Port is in the Idle state is asynchronous. If this bit is set, the assertion of the Request Output signals when the Port is in the Idle state is synchronous to chip select.

DO—Data Order

This bit defines the structure of the data bus and the order of bytes inside the 32-bit or 64-bit data structures. If DO = 0, then the byte order is most significant byte first (Motorola or IBM style). If DO = 1, then the byte order is least significant byte first (Intel or Digital style).

DW—Data Width

This bit selects the Port data width as follows:

DW	**Data Width**
0	32-bit
1	64-bit (Port A only)

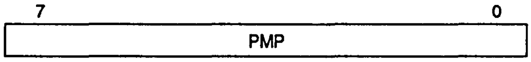
Note that if Port A is programmed for 64-bit data width, its control word is used for both Port A and Port B and Port A handles the most significant portion of the data, (bits 63-32).

PE—Port Enable

When this bit is zero, the operation of the Port is disabled and the Port remains in the Idle state. When this bit is set, the Port's request operation is enabled.

3.3.2 Port Memory Page Register (PMP) (IRT=8; IRI=6,7)

There are two Port Memory Page Registers (PMPs), one for each Port. (IRI = 6 for port A, and IRI = 7 for port B). The PMP Register specifies the external memory page length size utilized on a port, allowing the FSI to decide whether or not the next access is on the same page as the current access.



The Page Indication generated by the FSI may be very useful when the external memory is implemented by nibble or page mode dynamic memory in order to implement burst access and reduce bus bandwidth utilized by the FSI. The bits of the PMP, (7:0), specify the page length. The minimum page length is 16 bytes, (PMP = 0000 0000), and the maximum supported page length is 16k bytes, (PMR = 1111 1111). The port memory page length, in bytes, is specified in the following table.

PMP(7:0)	Length in Bytes
0000 0000	16
0000 0001	32
0000 0011	256
0000 0111	512
0000 1111	1k
0001 1111	2k
0011 1111	4k
0111 1111	8k
1111 1111	16k

3.3.3 Command Parameter Registers (CPR) (IRT=6; IRI=6,7)

There are two Command Parameter Registers, one for each Port's Command Register (IRI = 6 for port A, and IRI = 7 for port B). The CPR is used to define the source Port for the Transmit Command when it is to be issued directly to the FSI. The CPR operates similarly to the Ring Parameter Register with a Command Data Assignment (CDA) bit rather than a Ring Data Assignment (RDA) bit.

7	6	5	4	3	0
0	DPE	0	CDA	0	

DPE—Data Parity Enable

When DPE is set, parity checking is enabled for data transfers (DTR) from a descriptor in this command register.

CDA—Command Data Assignment\

When CDA is zero, the data for Transmit Commands issued directly to the FSI is read through Port A. When CDA is set, then it is read through Port B. In 64-bit operation this bit should be zero.

3.3.4 Ring Parameter Register (RPR) (IRT=6; IRI=0-5)

7	6	5	4	3	0
RPE	DPE	RPA	RDA	RML	

There are six Ring Parameter registers, one for each Ring (four transmit, and two receive rings). These registers define the Maximum Ring Length and select the Ports from which the Buffer Descriptor Rings and Data Buffers operate. The fields of these registers are described below:

RPE—Ring Parity Enable

When RPE is set, parity checking is enabled for ring entry transfers from this Ring.

DPE—Data Parity Enable

When DPE is set, parity checking is enabled for data transfers of this Ring.

RPA—Ring Port Assignment

When RPA is zero, the Ring entries are accessed through Port A. When RPA is set, the Ring entries are accessed through Port B. Note that a Ring may be assigned one Port space and its data may be assigned the other Port space. In 64-bit operation this bit should be zero.

RDA—Ring Data Assignment

When RDA is zero, transmit data for Rings 0, 1, 2, 3 or receive data for Rings 4 and 5 are transferred through Port A. When RDA is set, the data is transferred through Port B. In 64-bit operation this bit should be zero.

RML—Ring Maximum Length

These four bits specify the maximum length of the Ring, (the maximum number of entries with each entry being 64 bits). The following values of maximum length (in units of bytes) are defined:

RML	# of Entries	Maximum Length in Bytes
0	1	8
1	2	16
2	4	32
3	8	64
4	16	128
5	32	256
6	64	512
7	128	1k
8	256	2k
9	512	4k
10	1026	8k
11	2052	16k
12	4104	32k
13	8208	64k
14	16416	128k
15	32768	256k

For example, with an RML set to 3, i.e., 64 bytes, using 32-bit transfers, a current Ring address of 1BF8(hex) will be incremented to 1BFC (hex) and then on another increment will wrap to 1BC0 (hex).

3.3.5 Parameter Extension Register (PER) (IRT=7; IRI=0-7)

There are eight Parameter Extension registers, one for each channel. The PER has two exclusive definitions depending on whether the ring's secondary usage is as a Destination ring (for port to port DMA operations) or as a ring using the Local Memory option.

When it is desired to setup Destination rings for transmit channels (0-3) and for DMA Command Channels (6,7) these registers define the Destination Ring Maximum Length and its Ring and Data assignment as shown in Figure 4-12 (similar to RPR)

7	6	5	4	3	0
RPE	0	RPA	RDA	RML	

RPE—Ring Parity Enable

When RPE is set, parity checking is enabled for ring entry transfers from this Ring.

RP—Ring Port Assignment

When RPA is zero, the Ring entries are accessed through Port A. When RPA is set, the Ring entries are accessed through Port B. Note that a Ring may be assigned one Port space and its data may be assigned the other Port space. In 64-bit operation this bit should be zero.

RDA—Ring Data Assignment

When RDA is zero, DMA data is transferred through Port A. When RDA is set, the data is transferred through Port B. In 64-bit operation this bit should be zero.

RML—Ring Maximum Length

These four bits specify the maximum length of the Ring, (the maximum number of entries with each entry being 64 bits). The following values of maximum length (in units of bytes) are defined:

RML	# of Entries	Maximum Length in Bytes
0	1	8
1	2	16
2	4	32
3	8	64
4	16	128
5	32	256
6	64	512
7	128	1k
8	256	2k
9	512	4k
10	1026	8k
11	2052	16k
12	4104	32k
13	8208	64k
14	16416	128k
15	32768	256k

For example, with an RML set to 3, i.e., 64 bytes, using 32-bit transfers, a current Ring address of 1BF8 (hex) will be incremented to 1BFC (hex) and then on another increment will wrap to 1BC0 (hex).

For Transmit and Receive Channels (0-5) in Local Memory Mode these registers define Local Memory Space and its Port assignment and Parity control (see Figure 3-13). Note that starting addresses for each ring's memory space is defined by the Define Local Memory Command.

7	6	5	4	3	2		0
0	LPE	0	LPA	0	LML		

The fields in this case are described below:

LP—Local Memory Parity Enable

When LPE is set, parity checking is enabled for this Local memory space.

LPA—Local Memory Port Assignment.

When LDA is zero, the Local memory space is accessed through Port A. When LDA is set, the Local memory space is accessed through Port B.

LML—Local Memory Length

These three bits specify the length of the Local Memory Space for this channel. The following values of length (in units of bytes) are defined:

LML	Length in Bytes
0	8K
1	16K
2	32K
3	64K
4	128K
5	256K
6	512K
7	1M

3.3.6 Ring State Registers.(RSR) (IRT=E; IRI=0-7)

There are eight Ring State Registers (RSRs), one for each ring. For rings 6 and 7 only the DRY bit is provided.

7	6	5	4	3	2	1	0
EX	PER	OER	DRY	STP	RDY	EMP	CPL

EX—Exists

When EX is reset, this Ring does not exist in the system.

PER—Parity Error

PER is set when a Parity Error is detected by the FSI during a Ring entry's Read. This bit is reset when the Ring is redefined or transitioned to the Ready state by a Ring Ready control access (access to the FCR register).

OER—Operation Error

OER is set when a Port Operation Error or a First or Last bit error is detected by the FSI during a Ring entry's read. This bit is reset when the Ring is redefined or transitioned to the Ready state by a Ring Ready control access (access to the FCR register).

DRY—Destination Ring Ready

When DRY is set, this Destination Ring is in Ready state. This bit has meaning only if the ring is used for DMA operations. If used in Local Memory mode then this bit may be set or reset and has no meaning.

STP, RDY, EMP, CPL

The following table shows the meaning of bits in RSR in various Ring States:

Table 3-4. Ring Status Register Bit Values for Ring States

EX	STP	RDY	EMP	CPL	Ring State
0	x	x	x	x	Ring Does Not Exist
1	0	1	0	x	READY
1	1	0	x	x	STOP
1	0	0	1	0	EMPTY
1	0	0	x	1	COMPLETE

There are situations when more than one bit is set. For example, the RDY and CPL bits may be set if this Ring has been transitioned from the Complete state to the Ready state by a Ring Ready control access, but a descriptor read access has not yet been performed. Also, if a parity or operation error is detected during a descriptor read operation, the Ring is Empty (with no more valid descriptors), and the PER or OER is set in conjunction with the EMP bit. If EX is zero, the other bits have no meaning.

3.3.7 FIFO Watermark Register (FWR) (IRT=4; IRI=0-7)

There are eight FIFO Watermark Registers, one for each Ring and one for each Command Register. The Watermark for the Transmit Command issued directly to the FSI has the same meaning as in normal transmit operation.

7	0
FWM	

The Watermark has a different meaning for transmit and receive operations. In transmit operations, the Transmit Internal Data FIFO should reach the watermark or should hold an entire frame to be able to transmit data from a FIFO to assure the successful completion of transmission of the whole frame. Therefore, this number should be calculated according to the latency of the FSI external bus.

The value for the FIFO Watermark is calculated as follows: Watermark = (FWR)x32 bytes. The minimum value allowed for the FIFO Watermark is 64 bytes (FWR = 2), and the maximum value is 7360 bytes (FWR = 230).

In receive operations, the internal FSI Receive Data FIFO will either reach the Watermark or hold an entire frame before starting to transfer received data into external memory. Therefore, the calculation of the settings for the Receive Watermark Registers should normally be according to the number of memory cycles the system designer wants the FSI to use in DMA burst operation..

3.3.8 Limit Register (LMT) (IRT=5; IRI=0-5)

There are six Limit Registers, one for each Transmit and Receive Channel. The Limit register specifies the maximum number of frames allowed inside Local Memory Space for each channel.

7	6	0
0	LMT	

The minimum number of frames is 1 (LMT= 000 0000) and the maximum value is 128 (LMT= 111 1111)..

3.3.9 Receive Frame Type Registers (RFR) (IRT=1; IRI=4,5)

There are two Receive Frame Type Registers, one for each Receive Internal Data FIFO. The RFR defines the types of receive frames which can be received into each FIFO.

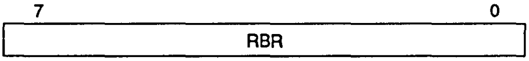
7	6	5	4	3	2	1	0
TE	0	OT	LS	LA	MF	SF	VF

With the RFRs, users may direct incoming frames to one of two Receive FIFOs; e.g., LLC frames to one FIFO, and MAC and SMT frames to the second FIFO. Each control bit in these registers defines whether a frame type is to be received into this FIFO, (when the bit is set), or not received into this FIFO, (when the bit is reset). If in both RFRs the bit is reset, frames with the indicated type will not be received by the FSI. If in both RFRs the bit is set, the frame is received only into Receive Data FIFO 4. The bit assignments are as follows:

Bit #	Bit Name	Frame Type
0	VF	Void frame
1	SF	SMT Frame
2	MF	MAC Frame
3	LA	LLC Asynchronous Frame
4	LS	LLC Synchronous Frames
5	OT	Other Frames
6		Reserved
7	TE	Token Cycle End indication from MAC . This mode cannot be used when using split header operation with Not Immediate header indication.

3.3.10 Receive Buffer Length Register (RBR) (IRT=3; IRI=4,5)

There are two Receive Buffer Length Registers, one for each Receive Ring. The RBR defines the length of the receive data buffers for each ring in external memory.



If the Receive Buffer Length register is used, the length is fixed for all buffers in the same Ring. If the buffer length is provided in the Receive Buffer Descriptor then this register should be set to zero. If Receive One Buffer mode is used then the Receive Buffer Length Register must be programmed. Only one bit in the RBR may be set to represent one of the eight buffer length choices as follows:

Bit #	Bit Name	Buffer Length
0	RBR0	64 bytes
1	RBR1	128 bytes
2	RBR2	256 bytes
3	RBR3	512 bytes
4	RBR4	1K bytes
5	RBR5	2K bytes
6	RBR6	4K bytes
7	RBR7	8K bytes

The largest Receive Data Buffer can hold the entire frame, (the FDDI standard maximum frame length is 4500 bytes). In 64-bit mode, the minimum allowed Receive Buffer Length is 128 bytes. Note that if the Receive Buffer Length Registers are used, the Header Length Register should not be programmed.

3.3.11 Header Length Register (HLR) (IRT=2; IRI=4)

There is one Header Length Register which defines the length of the frame header and it is used in header splitting operation.

Only one bit in the HLR may be set to represent one of the eight Header length choices as follows:

Bit #	Buffer Length
0	32 bytes
1	64 bytes
2	128 bytes
3	256 bytes
4	512 bytes
5	1K bytes
6	2K bytes
7	4K bytes

The minimum allowed Header Length is 32 bytes. Note that if the Header Length Register is used, the Receive Buffer Length Registers should not be programmed

3.3.12 Receive Memory Registers (RMR) (IRT=C; IRI=4,5)

There are two Maximum Receive Memory Space Registers, one for each Receive Ring. The RMR defines the maximum amount of internal memory to be allocated for the handling of the receive Ring.

Included in the RMR value are both the Receive Internal Ring FIFO and the Receive Internal Data FIFO. The value for the Maximum Receive Memory Space is calculated as follows: Max_Receive_Memory_Space = (RMR +1)x32 bytes. Therefore the largest is 8K bytes (RMR = 255). The minimum value of RMR should be 10 (Max_Receive_Memory_Space = 352 bytes)..

3.3.13 MACIF Transmit Control Register (MTR) (IRT=1; IRI=0)

7	6	5	4	3	2	1	0
TD3	TD2	TD1	TD0	TD7	TD6	0	TE

TE—Transmit Enable

When the TE bit is set, transmit operation is enabled. When TE is zero, transmit operation is disabled. During the transmit operation, if this bit is reset, the transmitter is disabled after the current transmission has been completed. This is a Global Transmit Enable bit.

TD(0-3,6,7)—Transmit Disable

There are six TD bits, one for each Transmit and Command Channel. When the TD bit is set, the transmit operation of this channel is disabled. When TD is zero, transmit operation of this channel is enabled. Note, however, that disabling one channel transmit operation will not affect other channels.

3.3.14 MACIF Receive Control Register (MRR) (IRT=1; IRI=1)

7	6	5	4	3	2	1	0
RPE	SLF	RMI	RAL	RCM	ROB/I HI	RE5	RE4

RPE—Receive Parity Enable

When RPE is set, the MACIF checks the incoming data parity according to the parity mode specified in the Port Control Register (PCR). If a parity error is detected, the MACIF generates the RABORT signal to the MAC and an appropriate indication is made for the affected frame. If RPE is zero, the MACIF does not check the incoming data parity. In this case, the parity is generated by the MACIF and is stored in the FSI internal memory with the associated data.

SLF—Split Frame

When SLF is set, the MACIF will split all received frames The Header portion of a frame will be directed to Channel #4. The rest of the data for this frame will be directed to one of the receive channels (4 or 5) according to its assignment specified in RFR register. The length of a header is specified in HLR register.

RMI—Receive My Frames Indications

When RMI is set, the MACIF receives an indication for frames which have been previously transmitted by this station. These indications are transferred to one of the Receive Rings according to its frame type.

RAL—Receive ALL

When RAL is set, the MACIF transfers all the data it receives to the internal memory, including remnant frames. This is useful in implementing a network analyzer. When RAL is zero, only valid frames are passed to the memory.

RCM—Receive CRC Mode

When RCM is set, the MACIF furnishes the received CRC with the received frame and the frame length includes four bytes of CRC. When RCM is zero, the CRC is not transferred to the internal memory and the frame length does not include the four bytes for the CRC. When SLF (Split Frame) or RAL (Receive ALL) is set, the CRC will always be furnished, and the RCM bit has no effect.

ROB/IH—Receive One Buffer or Immediate Header Indication

This bit has different meanings in Split and No Split operations.

In "No Split" operation this bit is ROB (Receive One Buffer). When ROB is set, the MACIF transfers only the first buffer of the frame into the internal memory . The buffer length is defined by the RBR register of the ring. However, the indication is furnished as in normal reception. When ROB is zero, the MACIF will transfer all the data of a received frame to the FSI internal memory.

In "Split" operation this bit is IHI (Immediate Header Indication). When IHI is set the header of the frame will be released immediately after its reception. When IHI bit is reset the header of the frame will not be released until the entire frame is received. Therefore, if the frame was aborted the header indication will be an error indication.

RE(5-4)—Receive Enable

When RE(i) is zero, the reception of data into the Receive FIFO(i) is disabled. In this case, the MACIF generates an RABORT signal to the MAC if the frame currently being received is directed to Receive FIFO(i).

3.3.15 Ring Ready Register (RDY) (IRT=0; IRI=0-5)

Ring Ready is used to transition the Ring specified by the IRI to the Ready state. The Ring Ready action is caused by an access to the register indicated by the IRT and IRI values. If this Ring was already in the Ready state, the Ring Ready control will have no effect. There are six Ring Ready Registers. The Ring Ready Access is not allowed to be performed on a Ring that does not exist in the System.

3.3.16 Destination Ring Ready (DRY) (IRT=9; IRI=0-3,6,7)

There are six Destination Ring Ready Registers, four for each transmit channel, or ring, and two for the DMA command rings. The Destination rings offer expanded capabilities and are programmed via the Parameter Extension Register (PER). Destination Ring Ready is used to transition the Destination Ring specified by the IRI to the Ready state. The DRY action is caused by an access to the register indicated by the IRT and IRI values. If this Ring was already in the Ready state, the DRY control will have no effect. The DRY Access is not allowed to be performed on a Ring that does not exist in the System.

3.3.17. Signal Register (SIG) (IRT=D; IRI=0-7)

These write-only registers are used in order to set one of the INT status bits inside Status Register 2 (SR2). Each SIG register is related to one INT bit when the IRI of this register defines the index of the bit. The INT(i) bit is set by writing SIG(i) register with predefined data as shown in Figure 3-23. Note, that INT bits in SR2 are sticky bits and may be cleared only by a host processor write access to the SR2 with the corresponding bit set.

7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0

3.3.18. User Register (USR) (IRT=F; IRI=4,5)

There are two USR read/write registers which may be used for Processor-to-Processor communication. The USR has no meaning for the FSI and it is not changed as a result of FSI operation.

3.3.19. FSI Revision Register (REV) (IRT=F; IRI=2)

This register is a read-only register which is used to identify the revision number of the FSI.

7	6	5	4	3	2	1	0
IFDDI	MRV			SRV			

IFDDI

When IFDDI=0 the FSI operates as stand alone device. When IFDDI=1 the FSI operates in the IFDDI device.

MRV—Major Revision

This field indicates the Major Revision Number of the FSI.

SRV—Sub Revision

This field indicates the Sub Revision Number of the FSI.

FSI Revision	**FSI Revision Register Value**
Revision A	0_000_0000
Revision B	0_001_0000
Revision C	0_001_0001
FSI Revision D stand alone (IFDDI in FSI Mode)	0_010_0000
FSI Revision D in the IFDDI (IFDDI in IFDDI Mode)	1_010_0000

3.3.20. Internal Error Status Register (IER) (IRT=F; IRI=0)

This register is a read-only register which is used to identify FSI internal operation errors. All bits are zero after reset, and are cleared, except MOV by a write access to the IER irrespective of the data being written. MOV is cleared only by a hardware or software reset.

7	6	5	4	3	2	1	0
IOE	IUE	TPE	MER	MOV	0	PBE	PAE

IOE—Internal Overrun Error

The MAC Interface experiences an internal overrun. During normal operation, the Main Controller assures a maximum response time for data transfers between the MAC Interface and the internal memory. If this response time exceeds these limits, this might indicate that the Main Controller is not performing properly. As a result, the Internal Overrun Error (IOE) may be set and the receiver is disabled (both RE4 and RE5 reset) and the RABORT signal is generated.

IUE—Internal Underrun Error

The MAC Interface experiences an internal underrun. During normal operation, the Main Controller assures a maximum response time for data transfers between the MAC Interface and the internal memory. If this response time exceeds these limits, this might indicate that the Main Controller is not performing properly. As a result, the internal Underrun (IUE) bit may be set, the current transmit frame is aborted with the appropriate indication and the transmitter is disabled (TE reset).

TPE—Transmit Internal Parity Error

The MACIF sets this bit when a parity error is detected on the data read from internal memory. In this situation the Transmit Parity Error (TPE) bit is set , the current transmit frame is aborted with an appropriate indication and the Transmit Enable (TE) bit in the MACIF Transmit Control Register (MTR) is reset.

MER—MAC Error

The FSI has detected a protocol handshake error between the MAC and the FSI during the reception of a frame. The MACIF will generate an RABORT to the MAC and will try to resynchronize with the MAC on the next frame. This error may be caused by some temporary problems (e.g., noise on control lines), therefore no special recovery functions are required. If there are permanent problems in the FSI to MAC interface (e.g., control lines are disconnected or forced to some constant value), the Host processor should perform an external diagnostic.

MOV—Memory Overrun

The FSI has experienced an internal memory overrun. This error may be caused by an incorrect definition of an internal FIFOs' Watermark. Therefore, the FSI parameter definitions should be checked and the FSI should be reinitialized.

PBE, PAE—Port A Internal Error. or Port B Internal Error

The Port A or Port B interface unit sets the appropriate bit when it detects a parity error while transferring data from the internal FSI memory buffer to the Port.

3.3.21 Software Reset (IRT=F; IRI=7)

Accessing the pseudo-register location "Software Reset" with the R bit of the FCR set will accomplish the same functions as activating the hardware RESET pin except that the Interrupt Mask Registers (IMR1 and IMR2) are unaffected by Software Reset. The CRF bit of Status Register 1 will be set when the Software Reset has been completed. This will generate an interrupt if the CFE bit of Interrupt Mask Register 1 (IMR1) is set.

SECTION 4
SIGNAL DESCRIPTIONS

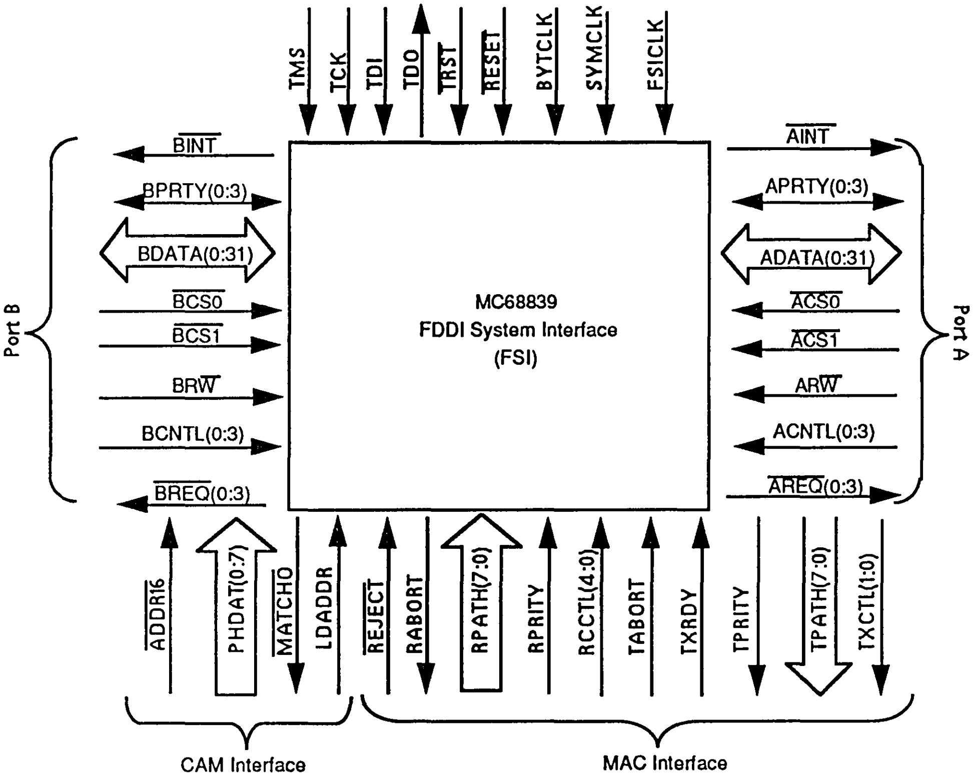


Figure 4-1. FSI Pinout

4.1 PORT A INTERFACE

The FSI has two identical ports for interfacing to a host system. These two ports allow the FSI to be used in various configurations. (see Appendix B)

Port A Data Bus (ADATA31-ADATA0)

This TTL-level, bidirectional, tri-state bus is used to read or write 32 bits of address, control, status or data from/into the FSI. This bus is asynchronous with chip clock and its timing is defined with reference to Chip Select signals ($\overline{\text{ACS0}}$, $\overline{\text{ACS1}}$).

Port A Parity (APRTY3–APRTY0)

These TTL-level, bidirectional signals indicate the parity of information presented on the ADATA bus. APRTY(0) is the parity of bits ADATA(7:0), and APRTY(3) is the parity of bits ADATA(31:24).

Port A Interrupt Request ($\overline{\text{AINT}}$)

This open-drain, active low output signal is used to notify an external processor of the occurrence of interrupting events. The interrupt level is asserted when both a status bit in the Status Register (SR1 or SR2) and its corresponding enable bit in the Interrupt Mask Registers (IMR1 or IMR2) are set. The $\overline{\text{AINT}}$ line changes its logical value synchronously with the internal clock. The user must supply a pull-up resistor for this signal to operate correctly.

Port A Request Lines ($\overline{\text{AREQ3}}$–$\overline{\text{AREQ0}}$)

These TTL-level active low output signals are used to indicate what type of data transfer is required by the FSI. $\overline{\text{AREQ}}$(3) defines the nature of the data. When $\overline{\text{AREQ}}$(3) is asserted, receive or transmit data transfers are requested. When $\overline{\text{AREQ}}$(3) is not asserted, then descriptor or indication information is to be transferred. The encoding of $\overline{\text{AREQ}}$(2:0) is as follows:

$\overline{\text{AREQ}}$(2)	$\overline{\text{AREQ}}$(1)	$\overline{\text{AREQ}}$(0)	
1	1	1	Idle State
0	1	1	Read Cycle
0	1	0	Read Cycle with address on the same memory page
1	0	1	Write Cycle
1	0	0	Write Cycle with address on the same memory page

Port A Control Lines (ACNTL3–ACNTL0)

These TTL-level input signals are used to select a register inside the FSI (see section 8.2.2). The address and data are treated as registers so that during data transfer cycles the ACNTL lines are assured by the system to determine whether addresses or data are being accessed. These lines should be asserted a set-up time before Chip Select assertion, and should remain stable at least hold time after Chip Select assertion.

Port A Chip Selects ($\overline{\text{ACS0}}$, $\overline{\text{ACS1}}$)

Two chip selects are provided for the configuration in which more than one master device is operating with the FSI. Normally, only one of these signals may be asserted for each access. These two TTL-level input signals are used to select one of the internal FSI registers according to the ACNTL combination. These signals are identical in their operation in 32- and 64-bit mode. $\overline{\text{ACS0}}$ and $\overline{\text{ACS1}}$ are the only reference signals to define the timing for other input or output signals for this port. When both chip selects are asserted, the Port Operation Error (POE) status bit is set to indicate the illegal operation. The external bus control logic may use this capability to indicate an external bus error.

Port A Read/Write (AR/W)

This TTL-level input signal is used to determine the direction of an access relative to the System Bus. When this line is low, data is written to the FSI. When this line is high, data is read from the FSI. This line should be asserted a set-up time before Chip Select assertion, and should remain stable at least hold time after the Chip Select assertion.

4.2. PORT B INTERFACE

Exactly the same as Port A except all signals have the prefix "B."

4.3. FSI/MAC INTERFACE

Transmit Data Bus (TPATH7–TPATH0)

Eight-bit, CMOS-level, output data bus for byte transfers from the FSI to the MAC. These lines are used at different times to carry either packet data, which could be either data to be sent or part of the packet request header, or extra control information.

Transmit Parity (TPRITY)

This CMOS-level output signal indicates the parity of the TPATH(7:0) data bus. The TXCTL lines are not included in the parity generation.

Transmit Control Signals (TXCTL1–TXCTL0)

These CMOS-level output signals are used by the FSI to indicate the type of data transfer. The encoding of these signals is as follows:

TXCTL(1:0)	Transfer Type
00	Filler
01	Tx_Start
11	Tx_Data
10	Tx_End

Transmit Ready (TXRDY)

This CMOS-level input signal, when asserted high, indicates to the FSI that the MAC is ready to accept additional TX_DATA transfer cycles. When the FSI detects that it is deasserted, low, it will continue to present the current TX_DATA cycle.

Transmit Abort (TABORT)

This CMOS-level input signal, when asserted, indicates to the FSI that it should abort the current transmission.

Receive Data Bus (RPATH7–RPATH0)

This is an eight-bit, CMOS-level input data bus for byte transfers from the MAC to the FSI. These lines are used at different times to carry either a pair of data symbols belonging to a received frame, status information describing the frame and why it ended, or packet frame status indicators.

Receive Parity (RPRITY)

This CMOS-level input signal indicates the parity of the RPATH(0:7) data bus. The RCCTL lines are not included in the parity calculation. The parity may be odd or even. The FSI may be programmed to check the parity or to ignore it. The correct parity is stored inside the FSI internal memory together with the associated data.

Receive Control Signals (RCCTL4–RCCTL0)

These CMOS-level input signals, which are controlled by the MAC, are used by the FSI to determine the type of MAC-to-FSI transfer. The RCCTL encoding definition is as follows:

RCCTL 43 210	Transfer Type
x0 000	Filler
00 101	Start Data
00 001	Data
x0 F11	End Data
xT F10	Frame Status
10 xxx	Token Cycle End
xx 100	Filler

In the above table, F is the flush/retain bit, and T is an indication that the frame status can be associated with a previously transmitted frame. This is further described in the MAC document.

Receive Abort (RABORT)

This CMOS-level output signal is used to indicate that the FSI will not accept any more data from the MAC. This signal is asserted by the FSI if there is no space in the receive internal data FIFO to store the information, if the data FIFO is disabled for reception, or if the $\overline{\text{REJECT}}$ signal is asserted when doing a CAM or external recognition logic match.

Reject Input Line ($\overline{\text{REJECT}}$)

This is a CMOS-level input signal to the FSI from external logic or an external CAM indicating that the FSI should reject the incoming frame. This signal would typically be used when the MAC is in promiscuous mode (MAC control signals are set as:
CopyAll = 00, CopyGroup = 1 and CopyIndLLC = 1).

Byte Clock (BYTCLK)\

This is a TTL-level input signal which has a frequency of 12.5 MHz and is synchronized with data byte transfers on both the transmit and receive data paths.

Symbol Clock (SYMCLK)

This is a TTL-level input signal which has a frequency of 25 MHz. This clock is the main FSI clock which is used for all internal synchronous operations in addition to data sampling on the FSI receive path.

4.4. CAM INTERFACE

PHY Receive Data Bus (PHDAT7–PHDAT0)

These eight CMOS-level input signals are connected to the MAC/ELM interface in order to provide an address matching function through the CAM inside the FSI operating on data currently being received by the MAC. PHDAT(3:0) should be connected to RCDAT(3:0) of the ELM circuit, and will contain the second symbol received of a symbol pair. PHDAT(7:4) should be connected to RCDAT(8:5) and will contain the first symbol received of a symbol pair. Note, that RCDAT(4) and RCDAT(9) are not connected to the FSI because they are always zero for data symbols during address or frame data reception.

Load Address (LDADDR)

This CMOS-level input signal is the main CAM control signal from the MAC. It is asserted active for only one BYTCLK cycle just before the first byte of the DA or SA field is presented on the PHDAT bus.

Long or Short Address ($\overline{\text{ADDR16}}$)

This input was maintained for backward compatibility. Since the MC68838 MAC chip will not copy a frame based on a 16-bit address match it is no longer a functional signal but should be tied off to ground or V_{CC} or left connected to the MC68838 $\overline{\text{ADDR16}}$ signal.

Address Match Line ($\overline{\text{MATHO}}$)

This open-drain output signal indicates to the MAC whether the address presented to the FSI CAM matches an individual or multicast (group) address stored in the CAM. It is asserted low upon match. This signal is examined by the MAC at the end of the received Source Address or Destination Address fields. When this signal is asserted, the received address belongs to this station's set of long addresses and the appropriate SA or DA actions should be taken by the MAC. The user must supply a pull-up resistor for this signal to operate correctly.

4.5. GENERAL I/O AND TEST

FSI Clock (FSICLK)

FSI internal logic clock pin which should be externally connected to SYMCLK.

Reset ($\overline{\text{RESET}}$)

This TTL input signal is internally sampled and synchronized by the FSI internal clock. When $\overline{\text{RESET}}$ is asserted, the FSI is initialized and placed in the reset state. This signal must be low for several clock periods to be recognized.

Test Clock (TCK)

This TTL input pin is the JTAG Clock. the TDO, TDI and TMS pins are synchronized by this pin.

Test Mode Select (TMS)

This TTL input pin is sampled by the FSI on the rising edge of TCK. This input signal is responsible for the state change in the Test Access Port State Machine. TMS must be held at '1' when the TRST signal changes from '0' to '1' When the TMS signal is not driven, an appropriate pull-up resistor must be connected to V_{CC}.

Test Data Input (TDI)

This TTL input pin is sampled by the FSI on the rising edge of the TCK. This input is the DATA to be shifted towards the TDO output.

When the Test Logic is enabled to serially shift the data, the information received at TDI will appear at TDO following a number of rising and falling edges of TCK determined by the length of the register selected (Instruction, Bypass or Boundary Scan Register).

When the TDI signal is not driven, an appropriate pull-up resistor must be connected to V_{CC}.

Test Data Output (TDO)

This TTL three-state output pin changes its logical value on the falling edge of the TCK. The capability of TDO to switch to a three-state drive allows parallel connection of board-level test data paths in cases where this is required.

Test Reset ($\overline{\text{TRST}}$)

This TTL input pin is the JTAG asynchronous reset. When asserted low, the Test Access Port is forced to the Test_Logic_Reset state and the FSI chip will work in its normal mode of operation. When $\overline{\text{TRST}}$ changes from '0' to '1', the TMS must be held at '1' in order to ensure deterministic operation of the Test Logic. If separate JTAG control of this signal is not required, i.e., JTAG is not implemented, then this signal should be tied to $\overline{\text{RESET}}$ or to GND.

SECTION 5
COMMANDS AND INDICATIONS

5.1 COMMANDS AND INDICATIONS OVERVIEW

The function of the FSI is to coordinate the transfer of data between its System Interface Ports and its MAC interface. Transfers of data, transmission and reception, are independent processes but rely upon similar principles and data structures. All data transfers are initialized and controlled by user supplied commands. The term Descriptor is reserved for a command which is a data buffer descriptor for either transmit or receive data buffers. The term command refers to all other descriptors or directly written commands which are used to alter the FSI's activity.

Both Ports have Command and Command Extension Registers; therefore, up to two Commands may be given directly to the FSI simultaneously. Note that Receive Buffer Descriptor Rings can contain only Receive Buffer Descriptors. Issuing commands through the use of Transmit Buffer Descriptor Rings allows multiple sequential commands to be provided to the FSI.

Indications are generated by the FSI after frame transmission or reception, and after command execution. These indications are then written back to the Buffer Descriptor Ring or to the Command Register. For transmit and receive frames, the frame indication status is placed inside the last buffer descriptor. No other descriptors will include frame status information.

5.1.1 Command/Descriptor/Indication

Each Command, Descriptor or Indication fits into a 64 bit longword. A template for a generic descriptor entry is shown in the following figure:

63	62	61	60	59 48
OWN	x	FIRST	LAST	x

47 32
x

31 16
x

15 0
x

All commands have the bits which are designated in Figure 5-1 in common, they are:

O Own bit. When set, the descriptor is available for use by the FSI.

bit 62 User Defined bit. Bit 62 may be written zero or one by the user. This bit has no meaning for the FSI and will not be changed when the indication is written back unless it is a "Command Error Indication" or a Receive Indication. Therefore the bit can be used as a semaphore for user defined applications.

First First Bit. When set, indicates that this is the first buffer of a frame. This is always set on non-data buffer descriptors.

Last Last Bit. When set indicates that this is the last data buffer for this frame. This is always set on non data buffer descriptors.

The remainder of the field is unique for each type of command. All commands which are not data buffer descriptors are single entries and therefore have the First and Last bits set. Those Data buffer descriptors which are pointers to buffers which are intended to hold an entire frame will also have the First and Last bits set.

Not all commands, indications or descriptors use the entire 64-bit field for information and/or data. However, the entire field must be provided when the command is issued as a Descriptor Ring entry. All indications are written out as 64-bit entities. When a 64-bit command is issued directly to the FSI, bits 31:0 are first written to the CER and then bits 64:32 are written to the CMR.

5.2 COMMANDS AND INDICATIONS

In cases where Commands are given directly to the FSI, the resulting indications are written into the respective Port's Command Register. Note that a new command cannot be issued until the corresponding indication has been written.

When a command read from a buffer descriptor ring is complete, its indication is written to that Ring and the Complete bit in the Status Register 1 (SR1) is set which causes an interrupt to the host if this interrupt is enabled. The next command from this Ring is executed after this previous command has completed its execution, (irrespective of whether the indication of the previous command's execution has been written to external memory).

The indications will always have the Own Bit (Bit 63) reset to indicate that the FSI has passed control of the descriptor entry to the host processor. Additionally, the First (Bit 61) and Last (Bit 60) bits are appropriately set: 10 in the first buffer of a multiple buffer frame, 11 in a single buffer frame, 01 in the last buffer of a multiple buffer frame, and 00 in intermediate buffers of a multiple buffer frame.

Table 5-1. FIRST and LAST Bit Settings

FIRST Bit	LAST Bit	Condition
1	1	Single Buffer Frame
1	0	First Buffer of a Multiple Buffer Frame
0	1	Last Buffer of a Multiple Buffer Frame
0	0	Intermediate Buffers of a Multiple Buffer Frame

5.2.1 General Commands and Indications

5.2.1.1 NOP Command

This command might be useful, for example, where frame entries inside the Transmit Buffer Descriptor Rings have aged too long. This command has to have both the First and Last bits set. The FSI will not create any activity executing this command except that the Command Done indication is given as for all commands. It should be emphasized that, in order to change the entries inside a Ring, the host processor should assure that the Ring is in the Stopped state.

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
1	x	1	1	1	1	1	1	0	0	0	0	0	0	0	0

47 — 32
x

31 — 16
x

15 — 0
x

5.2.1.2 Control Register Write Command

This command allows the user to write to any FSI internal control register.

63	62	61	60	59	58	57	56	55	54 — 51	50 — 48
1	x	1	1	0	1	1	0	0	IRT	IRI

47 — 32
x

31 — 24	23 — 16
x	DATA

15 — 0
x

IRT—Internal Register Type

See Section 3.2.12 on FCR definitions.

IRI—Internal Register ID

See Section 3.2.12 on FCR definitions.

Data—Internal Control Register Data

See Section 3.2.12 on FCR definitions.

The major advantage of using this command is in FSI initialization and when the user needs to change a control register on the fly as when writing "Ring Ready" commands. Instead of writing all the control registers by FCR accesses the user can prepare an initialization ring which includes control register write commands.

5.2.1.3 Move Command

This command moves 8 bytes of data between an external absolute memory address and an absolute internal (the 8K of FSI memory) address. The port used for the data transfer is determined by the RPA bit of the PER register along with the C bit as explained below. The user should not transfer data to the FSI internal memory without first reserving a block of that memory by use of the Get Block Command.

63	62	61	60	59	58	57	56	55	48
1	x	1	1	0	0	1	0	IBA	

47	46	45	34	33	32
0	IA		0	D	C

31	16
External Address High	

15	3	2	1	0
External Address Low		0		

IBA—Internal Block Address

The address of a block inside the FSI internal memory.

IA—Internal Address

The address of 8 byte word inside the block. (Each block is 32 bytes).

D—Direction

Direction of data transfer. When D is reset the data transfer occurs from the external memory to the FSI internal memory. When D is set the data transfer occurs from the FSI internal memory to the external memory.

C—Change Port

When C is reset, the external memory space has the same port assignment as indicated by the RPA bit in the PER register. When C is set, the external memory space is assigned to the port opposite that indicated by the RPA bit in the PER register.

5.2.1.4 Indirect Command

This command causes the FSI to take a command from an External memory and execute it. The external command will first replace the "Indirect" command inside the internal memory and then be executed as a normal command. Therefore, the indication of the External memory command will overwrite the original "Indirect" command. The port used to read the external command is defined by use of the RPA bit in the PER register along with the C bit as defined below.

63	62	61	60	59	58	57	56	55–48
1	x	1	1	0	0	1	1	0

47–33	32
0	C

31–16
External Address High

15–3	2–0
External Address Low	0

C—Change Port

When C is reset, the external memory space has the same port assignment as indicated by the RPA bit in the PER register. When C is set, the external memory space is assigned to the port opposite that indicated by the RPA bit in the PER register.

5.2.1.5 Get Block

This command causes the FSI to extract one 32 byte block of internal memory from the queue of free blocks. This block will no longer be used by the FSI, therefore it may be used for temporary storage by the user.

63	62	61	60	59	58	57	56	55	54	53	52	51–48
1	x	1	1	1	1	1	0	1	1	1	1	0

47–32
0

31–16
0

15–0
0

5.2.1.6 Get Block indication

This indication is generated in response to a Get Block command.

63	62	61	60	59	58	57	56	55	48
1	x	1	1	1	1	1	0	BNUM	

47	32
0	

31	16
0	

15	0
0	

BNUM—Block Number

The number of the block that is provided.and should be used as theInternal Block Address in Move commands.

5.2.1.7 Resource Request Command

The internal resource request and release commands can be used for synchronization between the activities of various channels. The Resource Request command requests the use of a single internal general purpose resource which is shared by all the DMA channels or rings. If the resource is occupied or used by another channel, the execution by the requesting channel will be delayed until the channel currently holding the resource releases it. If there are multiple Resouce Request Commands pending, the resource, when released, will be assigned according to the priority of the requesting channels: 6, 7, 0 1, 2 then 3.

63	62	61	60	59	58	57	56	55	54	53	52	51	48
1	x	1	1	1	1	1	0	1	1	0	0	0	

47	32
0	

31	16
0	

15	0
0	

5.2.1.8 Resource Release Command

The internal resource request and release commands can be used for synchronization between the activities of various channels. The Resource Release command causes this channel to release the internal general purpose resource.

<table>
<tr><td>63</td><td>62</td><td>61</td><td>60</td><td>59</td><td>58</td><td>57</td><td>56</td><td>55</td><td>54</td><td>53</td><td>52</td><td>51</td><td>48</td></tr>
<tr><td>1</td><td>x</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td colspan="2">0</td></tr>
<tr><td colspan="7">47</td><td colspan="7">32</td></tr>
<tr><td colspan="14">0</td></tr>
<tr><td colspan="7">31</td><td colspan="7">16</td></tr>
<tr><td colspan="14">0</td></tr>
<tr><td colspan="7">15</td><td colspan="7">0</td></tr>
<tr><td colspan="14">0</td></tr>
</table>

5.2.2 Ring Handling Commands and Indications

The Ring Handling Commands are used to define the Ring, to change Ring parameters and to change the Ring State. All commands of this type are single entry commands; therefore, both the First and Last bits should be set .

The Ring Pointer values are not necessarily the start of the external memory block which contains the ring. The Ring Pointer values indicate the next descriptor address to be accessed. Since the address wraps at the modulo value given in the Ring Parameter Register RPR (section 3.3.3), the Descriptor Ring memory block is defined as the pointer address modulo the RML value in the RPR.

5.2.2.1 Define Ring Command

This command may be issued at any time by the host processor through one of the Ports' Command Registers or may be placed inside one of the Transmit Buffer Descriptors Rings. Define Ring is used to define a new Transmit or Receive Ring or to change the definition of an existing Ring.

63	62	61	60	59	58	57	56	55	54	53	52	51	50	48
1	x	1	1	1	1	1	0	0	L	I	0	R	Ring#	

47	35	34	32
Ring Read Pointer Low		0	

31	16
Ring Read/Write Pointer High (Common)	

15	3	2	0
Ring Write Pointer Low		0	

Local (L)—Local Memory Use

This bit is set to indicate local memory use. I.e., it is expected that the user has implemented local memory on one of the ports to buffer data.

Indication (I)—Indication Generation

This bit is set when the user wants indications to be generated after the frame has been transferred to local memory for Tx rings 0, 1, 2 or 3. These indications do not indicate the transmission of the data, only the transfer of data prior to transmission.

Table 5-2. L(Local) and I(Indication) Bit Settings

L Bit	I Bit	Condition
0	0	Define normal ring (without using the local memory)
0	1	Invalid combination
1	0	Define ring that uses the Local Memory. The Indications will be generated after the transmission.
1	1	Define ring that uses the Local Memory. The Indications will be generated after the transfer of the frame to the Local Memory. This option may be used only for transmit rings (Ring# = 0,1,2 and 3)

Ready (R)—Ready State Transfer

If this bit is set, then the Ring will transition to the Ready state and attempt to read the first descriptor. If this bit is reset, this Ring will transition to the Complete state; i.e., the Ring is Empty.

Ring#—**Ring Number** (0-5)

Ring number, (0 - 5), of the Ring to be defined. The Ring number cannot be the Ring number of the Ring containing the command.

Ring Read Pointer Low

Ring Read Pointer Low Order start address in external memory.

Ring Read/Write Pointer High

This is the High Order address bits for the start address of both Read and Write pointers.

Ring Write Pointer Low

Ring Write Pointer Low Order start address in external memory.

The other parameters of a Ring, such as Ring Maximum Length (RML), Ring Port Assignment (RPA), and Ring Data Assignment (RDA) are defined by the Ring Parameter Register. In order to prevent uncertain operation, these parameters may be updated only when the Ring is Not Defined.

The Define Ring Command may be issued independently of the Ring's current state. However, the execution of this command is suspended until the Ring under definition is in the STOPPED, COMPLETE, or NOT_DEFINED states.

The region of the address space used for the ring does not necessarily begin with the Ring Pointer. Rather, the region is defined such that the most significant bits of the pointer remain constant. For example: Ring Pointer = 3445 4FF8 and an RML = 128 bytes (0100) provides an address range of 3445 4F80 to 3445 4FFC.

In normal Ring operation, the Read and Write pointers will have the same address at the start of the Ring. However, the capability to define different Read and Write pointers can be effectively used to enhance FDDI system operation. If, for example, the host processor wants to transmit a series of identical Beacon frames without the processor having to service each Beacon frame buffer descriptor, the user can define different Read and Write Pointer Low Addresses, and set the FIFO length such that they will not overlap. In this fashion, the updates for Read Pointers accessed and frames transmitted will not update the buffer descriptors, but are written to a different memory space, and the Own bit remains set in the transmit descriptors.

An example would be to set the Ring Read / Write Pointer High Address to FFFF, the Ring Read Pointer Low to FF00, the Ring Write Pointer Low to FF80, and to set the Ring Maximum Length to 32 bytes (4 buffer descriptors). A series of four buffer descriptors containing the frames to be continuously transmitted are placed starting at FFFF FF00, and the frames are continuously transmitted until the host processor stops the Ring.

If the user had defined the transmit ring with the option to provide an indication after the transfer of data from system to local memory, then the transmit indication has meaning only for that transfer. In this case the success or failure of transmission of data from local memory to the network will not be reported..

5.2.2.2. Set Destination Ring Command

This command may be issued at any time by the host processor through one of the Ports' Command Registers or may be placed inside one of the Transmit Buffer Descriptors Rings.

The Define Ring command (Define normal ring - L&I = 00) sets up the source ring and the Set Destination Ring sets up the Destination ring for the DMA transfers.

63	62	61	60	59	58	57	56	55	54	53	52	51	50 48
1	x	1	1	1	1	1	0	0	0	0	1	R	Ring#

47 35	34 32
Ring Read Pointer Low	0

31 16
Ring Read/Write Pointer High (Common)

15 3	2 0
Ring Write Pointer Low	0

R—Ready

If this bis is set, the Destination Ring will transision to the Ready state. If this bit is reset, the Destination Ring will remain in the Not Ready state.

Ring#—Ring Number (0–3, 67)

Ring number, (0 - 3, 6-7), of the Ring to be defined. The Ring number cannot be the Ring number of the Ring containing the command.

Ring Read Pointer Low

Ring Read Pointer Low Order start address in external memory.

Ring Read/Write Pointer High

This is the High Order address bits for the start address of both Read and Write pointers.

Ring Write Pointer Low

Ring Write Pointer Low Order start address in external memory.

The other parameters of the Ring, such as Ring Maximum Length (RML), Ring Port Assignment (RPA), and Ring Data Assignment (RDA) are defined by the Ring Parameter Extension Register. In order to prevent uncertain operation, these parameters may be updated only when the Destination Ring is Not Defined.

The region of the address space used for the ring does not necessarily begin with the Ring Pointer. Rather, the region is defined such that the most significant bits of the pointer remain constant. For example: Ring Pointer = 3445 4FF8 and an RML = 128 bytes (0100) provides an address range of 3445 4F80 to 3445 4FFC.

5.2.2.3 Set Local Memory Start Address Command

This command may be issued at any time by the host processor through one of the Ports' Command Registers or may be placed inside one of the Transmit Buffer Descriptors Rings.

The Set Local Memory Start Address Command specifies the start address in Local Memory and the Define Ring Command (Define ring that uses the Local Memory - L & I = 10 or 11) sets up the Transmit or Receive ring.

63	62	61	60	59	58	57	56	55	54	53	52	51	50 – 48
1	x	1	1	1	1	1	0	0	0	0	1	1	Ring#

47 – 35	34 – 32
x	0

31 – 16
Local Memory Start Address High

15 – 3	2 – 0
Local Memory Start Address Low	0

Ring#—Ring Number

Ring number, (0 - 5), of the Ring to be defined. The Ring number cannot be the Ring number of the Ring containing the command.

Local Memory Start Address High

This is the High Order address bits for the start address in external memory

Local Memory Start Address Low

This is the Low Order address bits for the start address in external memory.

The other parameters of the Ring, such as Local Memory Length (LML) and Local Memory Port Assignment (LPA) are defined by the Ring Parameter Extension Register. In order to prevent uncertain operation, these parameters may be updated only when the Ring is Not Defined.

The Set Local Memory Start Address Command should be issued before the Define Ring Command for the ring number under consideration. In normal practice, the registers pertaining to a given ring should be set up by the initialization routines and then the Set Local Memory Start Address Command can be issued to further delineate the memory area. Individual Define Ring Commands can then be issued as required by the application code.

The region of the address space used for the Local Memory does not necessarily begin with the Local Memory Start Address. Rather, the region is defined such that the most significant bits of the address remain constant. For example: Local Memory Start Address= 9454 3578 and an LML = 32K bytes (010) provides an address range of 9454 0000 to 9454 7FFC.

5.2.2.4 Using Define and set ring commands

1) Define Normal Ring will be done using Define Ring Command with L & I = 00.
2) Define Port to Port DMA channel will be done by the two commands:
   a) Set Destination ring command
   b) Define Ring Command with L&I = 00

   Note that the two commands should be issued in this order.
3) Define Port to Port DMA and normal transmission channel will be done by the two commands:
   a) Set Destination ring command
   b) Define Ring Command with L & I =00

   Note that the two commands should be issued in this order.
4) Define channel using the Local Memory will be done by the two commands:
   a) Set Local Memory Start Address command
   b) Define Ring command with L & I = 10 or 11

   Note that the two commands should be issued in this order

5.2.2.5 Stop Transmit Ring Command

The Stop Ring command may be issued at any time by the host processor through one of the Port's Command Registers or may be placed inside one of the Transmit Buffer Descriptors Rings.

63	62	61	60	59	58	57	56	55	54	53	52	51	50	48
1	x	1	1	1	1	1	0	0	0	1	0	0	Ring#	

47	32
x	

31	16
x	

15	0
x	

Ring#—Ring Number

Ring number, (0 - 5), of the Ring to be stopped.

The execution of this command is as follows:

a. When a frame is currently being transmitted from this Ring, its transmission is completed. If the current operation is a DMA data move, the transfer is continued until the end of the frame.

b. No more frames are transmitted from this Ring, even if they are already inside the FSI internal memory.

c. All indications prepared inside the FSI internal memory are written back to the Ring.

d. The Stop Ring Command is confirmed.

After this command has been completed, the host processor has an accurate status of all frames that have been transmitted from this Ring. However, the FSI internal memory related to a Stopped Ring may include one or more frames inside the FSI internal Transmit Data FIFO corresponding to this ring. If the host processor enables this Ring again, the FSI will continue operation from its current state.

In situations where the host processor decides to alter the Ring that has been stopped, the Define Ring Command for this Ring should be issued to provide new values for the .Ring Read/Write Pointers. All internal data previously associated with this ring will be cleared and lost.

Note that "Self Stop" (i.e., the target ring for the Stop command is its source ring) is also allowed. Although the ring is in the stop state, an indication is not generated immediately (the SR1 will have the appropriate RNR bit set) but is written when the ring is re-enabled. The indication is lost if the ring is re-defined

5.2.2.6 Read Ring Parameters Command

Execution of this command results in the return of a Read Ring Parameters Indication.

63	62	61	60	59	58	57	56	55	54	53	52	51	50	48
1	x	1	1	1	1	1	0	1	0	0	D	0	Ring#	

47	32
x	

31	16
x	

15	0
x	

Destination(D)—Destination Ring

When this bit is set, then this command is for Destination Ring. When D bit is zero the command is for the Source Ring.

Ring#—Ring Number

Ring number, (0 - 7), of the Ring whose parameters are to be read.

This command may be issued at any time by the host processor through one of the Ports' Command Registers or may be placed inside one of the Transmit Buffer Descriptors Rings.

5.2.2.7 Read Ring Parameters Indication

This indication is generated in response to a Read Ring Parameters Command. It includes the current Ring Read and Write Pointers, as shown below. All other parameters of the Ring may be read directly from the FSI internal registers.

63	62	61	60	59	58	57	56	55	54	53	52	51	50 48
0	x	1	1	1	1	1	0	1	0	0	D	0	Ring#

47 35	34 32
Ring Read Pointer Low	0

31 16
Ring Read/Write Pointer High (Common)

15 3	2 0
Ring Write Pointer Low	0

Destination (D)—Destination Ring

When this bit is set, then this command is for the Desintation Ring. When D bit is zero the command is for theSource Ring.

Ring#—Ring Number

Ring number, (0 - 7), of the Ring whose parameters are to be read.

Ring Read Pointer Low

Ring Read Pointer Low Order next descriptor read address in external memory.

Ring Read/Write Pointer High

This is the High Order address bits for the Read and Write pointer addresses.

Ring Write Pointer Low

Ring Write Pointer Low Order next descriptor write address in external memory..

5.2.2.8 Ring Reset Command

This command may be issued by the host processor through one of the Ports' Command Registers or may be placed inside one of the Transmit Buffer Descriptors Rings. Execution of this command causes the Ring to become Not Defined. All memory space occupied by this Ring in the FSI internal memory is released.

63	62	61	60	59	58	57	56	55	54	53	52	51	50	48
1	x	1	1	1	1	1	0	1	0	1	0	0	Ring#	

47	32
x	

31	16
x	

15	0
x	

Ring#—Ring Number

Ring Number, (0 - 5), of the Ring to become Not Defined.

5.2.3 Data Handling Commands and Indications

Data Handling Commands are used to transmit, receive and DMA transfer data frames. These commands are buffer descriptors, i.e., a pointer to an external memory buffer. A frame can be expected to occupy one or more descriptors. Transmit and DMA commands are normally placed in Transmit Buffer Descriptor Rings, and receive commands are placed in Receive Buffer Descriptor Rings. Note that transmit and DMA commands may also be issued through either Port's Command Register.

5.2.3.1 Transmit Buffer Descriptor Command

This command is used to describe a buffer of information to be transmitted. A Transmit frame is formed from up to 16 data buffers that are pointed to by one of the Transmit Buffer Descriptor Ring entries.

63	62	61	60	59	48
1	x	FIRST	LAST	0	

47	46	45	44	32
0	Tog	Mode	Buffer Length	

31	16
Buffer Address High	

15	0
Buffer Address Low	

Tog—Toggle

Toggle to the other port to be used as a source of data.

- **0** The data buffer will be taken from the same port as the previous data buffer.
- **1** The data buffer will be taken from the other port (This port will be used as a source of the data until another Toggle is encountered or until the end of the frame.

Using this option, a transmit frame may be constructed from buffers that are placed in different memory spaces (e.g. frame header may be prepared by the Local processor while the Data portion of the frame is taken from the System memory).

Mode—Mode selection

This field is appropriate for only the first descriptor of a frame.

- **0** Normal frame transmission.
- **1** Single frame transmit.

In Normal frame transmission the FSI will read the data of the next frame in the ring as long as there is sufficient internal memory available for storage. In single frame transmission only one frame at a time will be transferred to FSI internal memory and subsequently transmitted out the MAC interface. When the frame is transmitted then a subsequent single frame may be transferred into the FSI. This is valuable for very low performance bus applications. All the normal internal algorithms, descriptor rings, watermarks, etc., are in operation for the Single frame mode.

Buffer Length

The number of bytes in this buffer. Buffer Length, must not be zero.

Buffer Address

The byte address of the data buffer. Note that this address may have any byte alignment.

The first buffer descriptor of the frame has the First bit set and the final buffer descriptor for the frame has the Last bit set inside their respective Transmit Buffer Descriptor Commands. In all intermediate Transmit Buffer Descriptors, both the First and Last bits should be zero. In cases where a frame consists of only a single data buffer, then its buffer descriptor should have both the First and the Last bits set.

The data from the buffers is transferred by the FSI into one of its Internal Data FIFOs according to the source Ring Number. When this command is issued directly to the FSI using one of the Port Command Registers, the frame should consist of only a single data buffer with both the First and the Last bits set. The watermark must be set for Ring 6 when using Port A or Ring 7 when using Port B. The data is then placed inside the FSI Internal Command Data FIFO without affecting other Internal Data FIFOs.

All of the parameters specific to each frame; i.e., usable token type, method of transmission, etc. must be placed inside the first three bytes of the first data buffer. These parameters are called the Packet Header. They are transparent to the FSI and are forwarded to the MAC. They are fully described in the MC68838 MAC Specification.

5.2.3.2 Transmit Indication

This indication is generated after frame transmission and furnished in the last descriptor of the frame. The format for this indication is as following:

63	62	61	60	59 54	53 51	50	49	48
0	x	FIRST	1	0	PER	AB	UN	PE

47 32
Unchanged

31 16
Unchanged

15 0
Unchanged

F—First

This is the value of the First bit . This bit is set in the case of a single buffer frame.

PER—Port Error

This field specifies an error during data transfer through the FSI ports as described below:

PER	Error
001	Transfer to Data FIFO was aborted either by a Port Operation Error or an Abort Access.
010	Parity error during data transfer to the Data FIFO
100	Transfer from the IDF (intermediate Data FIFO) to Local memory was aborted either by a Port Operation Error or an Abort Access.
101	Transfer to the IDF was aborted either by a Port Operation Error or an Abort Access.
110	Parity error during data transfer to the IDF
111	Parity error during data transfer from the IDF to Local memory

AB—Abort

When AB is set, frame transmission has been aborted. When this bit is zero, transmission completed normally.

UN—Underrun

This bit is set when an underrun error caused a frame to be aborted. Subsequent transmit frames will, however, continue normally. Only the frame which is delimited by the descriptors when the underrun occurred is discarded.

PE—Parity ErrorTransmission of this frame was aborted due to a parity error.

Note that if the AB bit is set and UN, PE and PER are all zero, then the decision to abort frame transmission was caused by the assertion of the TABORT input signal. If the user had defined the transmit ring with the option to provide an indication after the transfer of data from system to local memory, then the transmit indication has meaning only for that transfer. In this case the success or failure of transmission of data from local memory to the network will not be reported..

5.2.3.3 DMA Buffer Descriptor Command

This command is used to transfer a buffer of information from a Source Ring to a Destination Ring. A DMA data frame is formed from up to 16 data buffers which are pointed to or described by the DMA Buffer Descriptor Ring entries. The Buffer length and Buffer Address fields are the same as used in the Transmit Buffer Descriptor Command. Note that Buffer Length field equal zero defines an 8 KBytes buffer.

DMA Buffer Descriptor Commands may be mixed with Transmit Buffer Descriptor Commands in the same ring that does not use the Local Memory Option. However, any Transmit Command preceding a DMA Command must be a Single frame transmit command (Mode = 1).

63	62	61	60	59 56	55 48
1	x	FIRST	LAST	0	INFO

47	46	45	44 32
1	Tog	1	Buffer Length

31 16
Buffer Address High

15 0
Buffer Address Low

INFO

This field is transferred into the DMA indication on the Destination side. It may be used in order to attach an ID to transferred data. If DMA frame is formed from several buffers, the INFO in the last descriptor is used.

5.2.3.4 DMA Indication (Source Side)

This indication is generated after frame transfer and furnished in the last source descriptor of the frame. The format for this indication is as following:

63	62	61	60	59 – 54	53 – 51	50	49 – 48
0	x	FIRST	1	0	PER	ER	0

47 – 32
Unchanged

31 – 16
Unchanged

15 – 0
Unchanged

F—First

This is the value of the First bit . This bit is set in the case of a single buffer frame.

ER—Error

This bit is set when the DMA transfer is aborted due to error.

PER—Port Error

This field specifies an error during data transfer through the FSI ports as described below:

PER	Error
001	Transfer to Data FIFO was aborted either by a Port Operation Error or an Abort Access.
010	Parity error during data transfer to the Data FIFO
100	Transfer from the Data FIFO to an Destination Buffer was aborted either by a Port Operation Error or an Abort Access or Discard Frame (Descriptor with D=1)
111	Parity error during data transfer from the Data FIFO to a Destination Buffer

5.2.3.5 Destination Buffer Descriptor

This descriptor is used to describe a Destination Data Buffer available to hold the DMA'ed data. The Destination Buffer Descriptor is placed only in the destination ring. This descriptor is identical to the Receive Buffer Descriptor and is repeated here for convienence.

Bits	Field
63	1
62	x
61–48	0
47–45	0
44–37	Buffer Length
36–32	0
31–16	Buffer Address High
15–2	Buffer Address Low
1	0
0	D

Buffer Length

The buffer length may be given for each destination descriptor. There are eight possible length as listed below:

Buffer Length Field	Buffer Length
0000.0001	64 bytes
0000.0010	128 bytes
0000.0100	256 bytes
0000.1000	512 bytes
0001.0000	1K bytes
0010.0000	2K bytes
0100.0000	4K bytes
1000.0000	8K bytes

Note, that buffer length should be specified for Destination Buffer Descriptors.

Buffer Address

The address of the first byte in the associated data buffer. Note that this address must be aligned on longword (64-bit) boundaries for 64-bit accesses and aligned to 32-bit boundaries for 32-bit accesses.

D—Discard Frame.

If D is set then the rest of current DMA frame is discarded, that is, the DMA transfer is aborted for this frame.

5.2.3.6 DMA Indication Without Error (Destination Side)

This indication is generated after frame transfer and furnished in the last destination descriptor of the frame. The format for this indication is as following:

63	62	61	60	59 ... 56	55 ... 48
0	x	FIRST	1	0	INFO

47	46	45	44 ... 32
Cmd			Frame Length

31 ... 16
Unchanged

15 ... 0
Unchanged

INFO

This field is a copy of the same field in last source descriptor.

Cmd

This field is a copy of the same (bits 47,46,45) field in last source descriptor.

Frame Length

The length of entire frame. modulo 8K.

5.2.3.7 DMA Error Indication (Destination side)

This indication is generated after frame transfer and furnished in the last destination descriptor of the frame. The format for this indication is as following:

63	62	61	60	59	58	57	56	55	48
0	x	FIRST	1	1	0	0	1	INFO	

47	46	45	44	35	34	33	32
Cmd			0		PER		

31	16
Unchanged	

15	0
Unchanged	

INFO

This field is a copy of the same field in last source descriptor.

Cmd

This field is a copy of the same (bits 47,46,45) field in last source descriptor.

PER—Port Error

This field specifies an error during data transfer through the FSI ports as described below:

PER	Error
001	Transfer from a Source memory to an Internal memory was aborted either by a Port Operation Error or an Abort Access.
010	Parity error during data transfer from a Source memory to an Internal memory
100	Transfer from an Internal memory to a Destination Buffer was aborted either by a Port Operation Error or an Abort Access or Discard Frame (Descriptor with D=1).
111	Parity error during data transfer from an Internal memory to a Destination Buffer

5.2.3.8 Make Indication Command

This command is used to transfer a 24-bit data word, the Indication field, to a Destination Ring. The Indication field of this command will be written to the same field inside the Descriptor in a Destination Ring.

63	62	61	60	59	58	57	56	55 48
1	x	1	1	0	0	0	1	Indication

47 32
Indication

31 16
x

15 0
x

Indication

The indication field of this command will be written to the same field inside the destination descriptor.

5.2.3.9 Indication (Destination side)

This indication is generated on a Destination side as a result of Make Indication command from source. It is written to a Destination Buffer descriptor. The user is cautioned that if Make Indications are mixed with DMA Commands the indications are indistinguishable in the case of error.

63	62	61	60	59	58	57	56	55 48
0	0	1	1	1	0	0	1	Indication

47 32
Indication

31 16
Unchanged

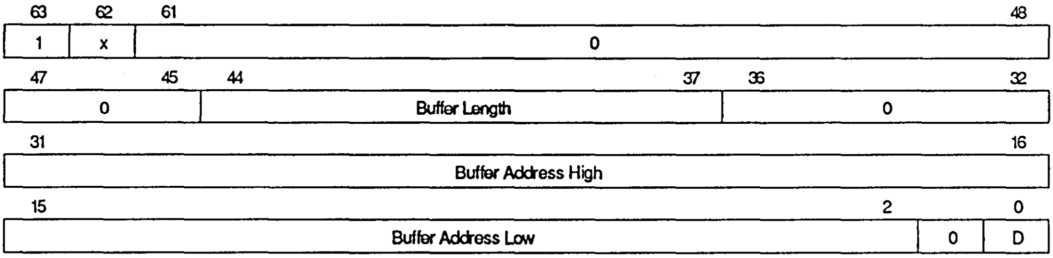
15 0
Unchanged

Indication

The indication field is a copy of the same field in Make Indication command.

5.2.3.10 Receive Buffer Descriptor

This descriptor is used to describe a Receive Data Buffer available to hold the received data. This descriptor is identical to the Destination Buffer Descriptor and is repeated here for convienence.



Buffer Length

The buffer length may be given for each descriptor. There are eight possible length as listed below:

Buffer Length Field	**Buffer Length**
0000.0001	64 bytes
0000.0010	128 bytes
0000.0100	256 bytes
0000.1000	512 bytes
0001.0000	1K bytes
0010.0000	2K bytes
0100.0000	4K bytes
1000.0000	8K bytes

If the Buffer Length Field is zero, then Receive Buffer Length specified by RBR internal register is used by the FSI.

Buffer Address

The address of the first byte in the associated data buffer. Note that this address must be aligned on longword (64-bit) boundaries for 64-bit accesses and aligned to 32-bit boundaries for 32-bit accesses.

D—Discard Frame

If D is set then the rest of current Receive frame is discarded.

5.2.3.11 Receive Frame Normal Indication

This indication is generated when a frame has been received normally. Note that this frame could still have a CRC error. The FNUM, EF, AF, CF, F0, F1 fields are taken from Frame Status information received from the MAC. The CE and DA fields are taken from the End Data indication also provided by the MAC. These fields are further described in the MC68838 MAC document and are known as the C and DD fields respectively.

63	62	61	60	59	58	57	56	55 53	52	51	50	49	48
0	S	FIRST	1	0	0	0	CE	FNUM	EF	AF	CF	F0	F1

47 46	45	44 32
DA	0	Frame Length

31 16
Unchanged

15 0
Unchanged

S—Split

When a frame is split into a Header portion and a Data portion each one of these parts will be treated by the FSI as a whole frame. In order to distinguish between these frames and normal frames the S bit in the indication (both for Header and Data) will be set. For frames that are not split this bit will be reset.

F—First Bit

0 for a multiple buffer frame and 1 for a single buffer frame.

CE—CRC Error

FNUM—Number of Valid Flags

EF—E Flag

AF—A Flag

CF—C Flag

F0—First Additional Flag

F1—Second Additional Flag

DA—Destination Address Match Field

1 1	Local Match
0 1	External CAM Match
1 0	Promiscuous reception
0 0	Reserved

Frame Length

This field holds the length of a frame. When a frame is received in header split mode the Header indication will have a Length field equal to the Header Length and the Data indication will have a length field equal to the length of the data only.

Note: When using split mode the status fields CE, FNUM, EF, AF, CF, F0, F1 and DA are not valid in the header indication..

5.2.3.12 Receive Error Indication

This indication is generated by the FSI when an error, fragment, or secondary NSA frame, (Next Station Address frame with the A bit set), is received when the FSI is in the Receive ALL (RAL) mode (MACIF Receive Control Register (MRR)), the receive operation has been aborted by the FSI, or when the fragment is longer than receive Watermark. Otherwise, under normal operation, frames with these errors are neither seen by the FSI, nor reported in the Receive Error Indication except for Secondary NSA frames.

63	62	61	60	59	58	57	56	55 53	52	51	50	49	48
0	S	FIRST	1	0	1	OE	PE	FNUM	EF	AF	CF	F0	F1

47 45	44 32
STT	Frame Length

31 16
Unchanged

15 0
Unchanged

S—Split

When a frame is split into a Header portion and a Data portion each one of these parts will be treated by the FSI as a whole frame. In order to distinguish between these frames and normal frames the S bit in the indication (both for Header and Data) will be set. For frames that are not split this bit will be reset.

F—First Bit

0 for a multiple buffer frame and 1 for a single buffer frame.

OE—Overrun Error

Overrun Error in the FSI

PE—Parity Error

Detected by the MACIF on the receive path.

FNUM—Number of Valid Flags

EF—E Flag

AF—A Flag

CF—C Flag

F0—First Additional Flag

F1—Second Additional Flag

STT—Receive Status of MAC.

The STT field is derived from the end data information from the MAC except the 000 case which in a Receive Error indication is used to indicate receipt of a Secondary NSA frame.

111	MAC Reset
110	Format Error
101	Fragment
100	Invalid Length
011	FSI Abort
010	DA Not Matched
001	SA Matched
000	Secondary NSA Frame

Frame Length

This field holds the frame length of a frame. When a frame was split, the Header portion of a frame will have a Frame Length equal to the Header Length and a Data portion of a frame will have a frame length equal to a length of this portion only.

NOTES:

When using split mode the status fields FNUM, EF, AF, CF, F0, F1 and STT are not valid in the header indication.

In the event of a MAC interface error (MER) as reported in the IER register, bits 55 to 48 of this indication will be zero and the STT field is not valid nor meaningful. The FNUM, EF, AF, CF, F0, F1 fields are taken from Frame Status information received from MAC. This is further described in the MC68838 (MAC) document..

5.2.3.13 Receive Port Error Indication

The Receive Port Error Indication is provided as a result of an error during data transfer through one of the FSI's ports.

63	62	61	60	59	58	57	56	55 53	52	51	50	49	48
0	S	F	1	1	0	0	1	FNUM	EF	AF	CF	F0	F1

47 46 45	44 36	35 32
Not Specified	0	PER

31 16
Unchanged

15 0
Unchanged

S—Split

When a frame is split into a Header portion and a Data portion each one of these parts will be treated by the FSI as a whole frame. In order to distinguish between these frames and normal frames the S bit in the indication (both for Header and Data) will be set. For frames that are not split this bit will be reset.

F—First Bit.

0 for a multiple buffer frame and 1 for a single buffer frame.

OE—Overrun Error

Overrun Error in the FSI

PE—Parity Error

Detected by the MACIF on the receive path.

FNUM—Number of Valid Flags

EF—E Flag

AF—A Flag

CF—C Flag

F0—First Additional Flag

F1—Second Additional Flag

PER—Port Error

This field specifies an error during data transfer through the FSI ports as described below:

PER	Error
1100	Transfer from Data FIFO was aborted either by a Port Operation Error or an Abort Access or Discard Frame (Descriptor with D=1).
1111	Parity error during data transfer from the Data FIFO
0100	Transfer from the IDF (intermediate Data FIFO) to System memory was aborted either by a Port Operation Error or an Abort Access or Discard Frame (Descriptor with D=1).
0101	Transfer to the IDF was aborted either by a Port Operation Error or an Abort Access.
0110	Parity error during data transfer to the IDF
0111	Parity error during data transfer from the IDF to System memory

5.2.3.14 SPLIT MODE DATA ERROR INDICATION

The Split Mode Data Error Indication is generated when split mode is enabled and an error occurs on the data portion of the frame which causes a discard of the entire data portion. This indication is generated to provide a means of synchronization between the header ring and the data ring so that for every header indication there will be a corresponding data indication even in the event of an error. If the received data had already been partially or completely transferred and an error occurs the result will be a Receive Error Indication as described above.

63	62	61	60	59	58	57	56	55	53	52	51	50	49	48
0	0	1	1	1	0	1	0	0						

47	46	45	32
0			

31	16
Unchanged	

15	0
Unchanged	

5.2.3.15 Receive My Frame Indication

This indication is generated if enabled through the RMI bit in MACIF Receiver Control Register of the MAC. It indicates that a frame transmitted by This Station has been received and stripped. Note that this indication is placed inside one of the Receive Descriptor Rings according to the received frame's Frame Control field. The FNUM, EF, AF, CF, F0, F1 fields are taken from Frame Status information received from MAC. This is further described in the MC68838 MAC document.

63	62	61	60	59	58	57	56	55 53	52	51	50	49	48
0	0	1	1	1	0	0	0	FNUM	EF	AF	CF	F0	F1

47 46	45 32
SA	0

31 16
Unchanged

15 0
Unchanged

FNUM—Number of Valid Flags

EF—E Flag

AF—A Flag

CF—C Flag

F0—First Additional Flag

F1—Second Additional Flag

SA—Source Address Match Field

1 1	Local match
1 0	External Match
0 1	Bridge Match
0 0	The MAC aborted the frame prior to an SA match.. Refer to the MAC MC68838 manual on Frame Status indications..

5.2.3.16 Token Cycle End Indication

This indication is generated if enabled by the TE bit inside one of the Receive Frame Type Registers when the MAC chip issues a Token Cycle End event and the FSI has accomplished at least one frame transmission during the current Token Cycle. The format of this indication is as follows:

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0

47	40	39	32
0		Frame Counter	

31	16
Unchanged	

15	0
Unchanged	

Frame Counter

The Frame Counter indicates the number of frames that were transmitted and not received back. This number should normally be zero.

5.2.4 CAM Commands

5.2.4.1 Set up CAM Command

This command may be issued by the host processor through one of the Ports' Command Registers or may be placed inside one of the Transmit Buffer Descriptors Rings.

63	62	61	60	59	58	57	56	55	54	53	52	48
1	x	1	1	1	0	0	1	V	0	0	CAM_ID	

47	32
CAM Entry	

31	16
CAM Entry	

15	0
CAM Entry	

V—Valid

0 for invalid entry, 1 for a valid entry.

CAM_ID—FSI CAM entry address.

CAM Entry Address Entry

48-bit address entry to place inside the CAM at CAM_ID.. Bit 47 is the I/G bit.

5.2.4.2 Read CAM Entry Command

This command is used to read particular CAM entries. This would be done for CAM device testing, etc. The value of the CAM entry being read is returned in the indication of the Read CAM Entry command.

63	62	61	60	59	58	57	56	55	54	53	52	48
1	x	1	1	1	0	1	0	0	0	0	CAM_ID	

47	32
x	

31	16
x	

15	0
x	

CAM_ID—FSI CAM entry address

Bit 47 is the I/G bit.

This command may be issued by the host processor through one of the Ports' Command Registers or may be placed inside one of the Transmit Buffer Descriptors Rings.

5.2.4.3 Read CAM Entry Indication

This indication is generated as a response to the Read CAM Entry Command.

63	62	61	60	59	58	57	56	55	54	53	52	48
0	x	1	1	1	0	1	1	V	0	0	CAM_ID	

47	32
CAM Entry	

31	16
CAM Entry	

15	0
CAM Entry	

V—Valid

0 for invalid entry, 1 for a valid entry.

CAM_ID—FSI CAM entry address.

CAM Entry—Address Entry

48-bit address entry inside the CAM at CAM_ID. Bit 47 is the I/G bit.

5.2.4.4 Compare CAM Entry Command

This command is used for testing the CAM device and/or to determine that the CAM has been programmed correctly.

<table>
<tr><td>63</td><td>62</td><td>61</td><td>60</td><td>59</td><td>58</td><td>57</td><td>56</td><td>55</td><td>54</td><td>53</td><td>52</td><td>51</td><td>50</td><td>49</td><td>48</td></tr>
<tr><td>1</td><td>x</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>47</td><td colspan="14"></td><td>32</td></tr>
<tr><td colspan="16">CAM Entry (47–32)</td></tr>
<tr><td>31</td><td colspan="14"></td><td>16</td></tr>
<tr><td colspan="16">CAM Entry (31–16)</td></tr>
<tr><td>15</td><td colspan="14"></td><td>0</td></tr>
<tr><td colspan="16">CAM Entry (15–0)</td></tr>
</table>

CAM_ID—FSI CAM entry address.

CAM Entry—Address Entry

48-bit address entry to compare with CAM entries. Bit 47 is the I/G bit.

This command may be issued by the host processor through one of the Ports' Command Registers or may be placed inside one of the Transmit Buffer Descriptors Rings.

5.2.4.5 Compare CAM Entry Indication

This indication is generated in response to a Compare CAM Entry Command.

<table>
<tr><td>63</td><td>62</td><td>61</td><td>60</td><td>59</td><td>58</td><td>57</td><td>56</td><td>55</td><td>54</td><td>53</td><td>52</td><td>51</td><td>50</td><td>49</td><td>48</td></tr>
<tr><td>0</td><td>x</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>M</td><td>1</td><td>0</td><td>0</td><td colspan="5">CAM_ID</td></tr>
<tr><td>47</td><td colspan="14"></td><td>32</td></tr>
<tr><td colspan="16">Unchanged</td></tr>
<tr><td>31</td><td colspan="14"></td><td>16</td></tr>
<tr><td colspan="16">Unchanged</td></tr>
<tr><td>15</td><td colspan="14"></td><td>0</td></tr>
<tr><td colspan="16">Unchanged</td></tr>
</table>

M—Match Indicator

0 = No match, 1 = match.

CAM_ID—FSI CAM entry address.

If the M bit is set, the entry has been found in the FSI CAM and its ID is indicated in the CAM_ID field. If there is no match to any FSI CAM entry, the M bit is zero and the CAM_ID is not valid.

SECTION 6
FUNCTIONAL OPERATION

The Function of the FDDI System Interface device is the transfer of frame data between the system bus/memory and the FDDI Media Access Controller. The FSI may also serve as a DMA engine and transfer data from one memory area to another
This chapter serves as an overview for the data DMA functionality and as an introduction to the data types used with the Motorola FSI. The chapter on Programming the FSI has more detailed information on realizing the functionality introduced here.

6.1 DMA FUNCTIONAL OVERVIEW

This section illustrates some of the possible configurations for the transfer of data between the system interface and the FDDI interface or for data transfers from one memory area to another over the system port interface(s).

6.1.1 Direct Transmit and Receive Operation

Figure 6-1 shows the simplest and most immediate method for the transmission and reception of frame data to and from the FDDI interface.

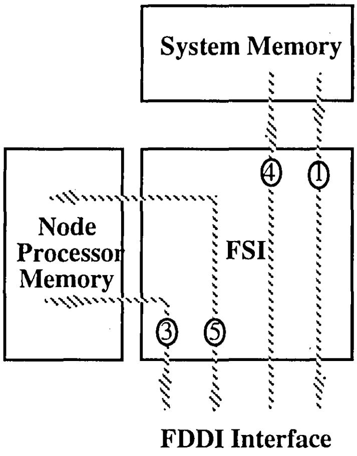


Figure 6-1. Direct Transmission and Reception

Figure 6.1 shows the use of both ports A and B. However, the implementor may desire the use of only one port, in which case, the idle port could be used to expand the interface to 64-bit multiplexed address/data operation, provide a non-multiplexed address source or one of the ports may simply be left idle.

In the direct transmit and receive implementation the user may use up to four transmit channels, or rings, numbered 0-3, and two receive channels, numbers 4 and 5. The ring numbers are fixed however, they can be individually assigned to either port via the Ring Parameter Registers.

In figure 6-1 Tx ring 1 and Rx ring 4 are assigned to the port on the system interface and Tx ring 3 and Rx ring 5 are assigned to the port on the local memory system.

6.1.2 Transmit and Receive Operation with Memory Expansion

The FSI provides an internal 8K FIFO buffer for frame data. This internal memory provides a latency buffer between the FDDI interface and the systems memory. The 80 usecs per KBytes of bus latency provided by the internal 8K memory is still insufficient for many applications which have slow or very active system bus interfaces. It is possible to expand the size of the latency buffer through the use of external memory using a local memory option. Such an implementation is shown in figure 6-2.

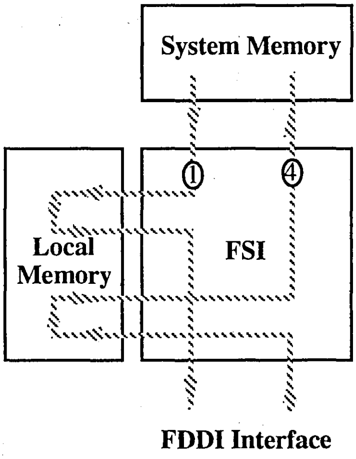


Figure 6-2. Extended Local Memory Option

In this configuration, transmit data from the system is first transferred to the local or extended memory. When an entire frame has been transferred the frame data will be available to be transferred to the FDDI interface. Receive data from the FDDI interface is first put into local memory and will subsequently be transferred to system memory. Note that the local memory is assumed to be readily available to the FSI with low system bus contention. This insures that frame data is transferred between the local memory and the FDDI interface in a timely fashion. The transfer of data between the local memory and system memory may then take place at a slower pace determined by the amount of contention on the system bus port.

In figure 6-2 we have portrayed only two channels one receive and one transmit both using the expanded memory. All of the transmit and receive channels may make use of the expanded memory and even mixed use is possible. Figure 6-3 shows two channels using the expanded memory and two channels using the direct transfer method.

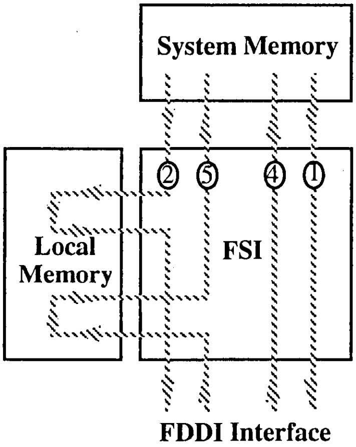


Figure 6-3. Direct and Expanded Local Memory Options

The local memory may also be shared by a local processor and have specific channels assigned to it. However, the implementor must insure that the local memory interface provides sufficient bandwidth to handle the data flow. Such an application is shown in figure 6-4.

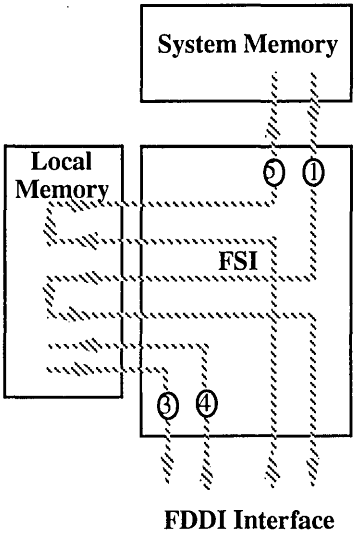


Figure 6-4. Dual Ports with Local Memory Option

Figure 6-4 shows the assignment of Tx ring 1 and Rx ring 5 on the System memory port, implemented perhaps due to a low bus latency. Tx Ring 3 and Rx ring 4 are assigned to the local bus using direct transfer.

6.1.3 Memory to Memory DMA Options

The FSI also provides facilities to use the DMA channels, or rings, for the purpose of transferring data from one memory location to another even across ports. Figure 6-5 shows the variety of data DMA transfers of which the FSI is capable.

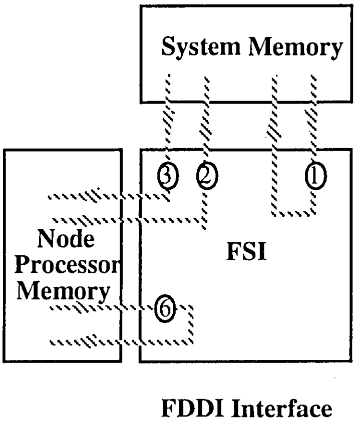


Figure 6-5. Memory to Memory DMA

Figure 6-5 only shows memory to memory DMA operations. The Memory DMA operations may take place simultaneously with any of the FSI's transfer of data from either port's memory and the FDDI interface. Memory to memory DMA operations may be used to transfer any block of data. For example, host processors on one port may use this facility to download data or code to another processor's memory space on the other port.

Any transmit ring (0-3), not using local memory, or Command channel (6-7) may be used for DMA operations as well as frame data transmission to the FDDI interface.

6.2 FSI DATA STRUCTURES

The primary purpose of the FSI is to facilitate the transfer of frame data packets between the system memory and the FDDI interface, namely the FDDI MAC device. This subsection defines the nature of the frame data and what the user needs to know about the memory format of the data and about the data structures which the FSI uses to effect the data transfer. When the FSI is used to perform memory to memory DMA data moves the user may disregard the nature of the data since that data may not necessarily be destined for the FDDI interface.

The external memory frame data can exist in one contiguous buffer or can be broken up into several buffers. Each buffer in memory is accessed by the FSI through the use of a Descriptor. Descriptors may be thought of as commands, commands to receive or transmit frame data, or in the case of performing a memory to memory move, a DMA data command.

Descriptor Rings are conveniently organized ring structures which are collections of Descriptors and Commands. A descriptor ring may be conveniently thought of as defining a DMA channel. Commands which deal with setting up and manipulating Descriptor Rings are dealt with in detail in section 5.2.1. There are two receive Descriptor Rings (numbers 4 and 5) which contain only Receive Buffer Descriptors and four transmit Descriptor rings (numbers 0, 1, 2 and 3) which can contain general commands and Transmit Buffer Descriptors.

In addition, two Command channels (numbers 6 and 7) are provided for the use of general commands, transmit frame commands, DMA commands for memory to memory data moves but not receive descriptor commands.

6.2.1 FDDI Frame Structure

The basic data structure in FDDI is the frame. The FDDI frame structure that the user is most involved with is depicted as:

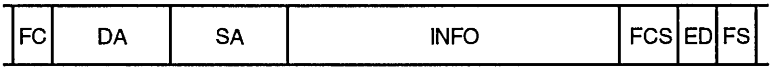


Figure 6-6. FDDI Frame Structure

The length of the various fields is: Frame Control (FC) field is 1 byte, Destination Address (DA) field 2 or 6 bytes, Source Address (SA) field 2 or 6 bytes, the INFO or data field is of variable length and the Frame Check Sequence (FCS) field contains the Cyclic Redundancy Check (CRC) and is 4 bytes in length. The End Delimiter (ED) and Frame Status Fields are together a minimum of two bytes. Interpacket Idle symbols are not depicted. The MAC chip automatically provides the FCS, ED and FS fields as well as the interpacket Idle stream. The user must supply the FC, DA, SA and INFO fields. Bridge applications have the option of supplying the CRC for the FCS field and suppressing the CRC generation by the MAC for data integrity.

Within system memory the user data fields might look as shown in figure 6-7. Here, two examples are shown, Frame 1, in which the entire frame is contained in one buffer, and Frame 2 in which the frame is segmented into three buffers.

6.2.2 Example Memory Organization

First a few notes on Figure 6-7 are in order. The three bytes of Packet Header are shown here as they contain control information for the MAC. These must be supplied by the user and are fully described in the MC68838 MAC manual. The FCS field is shown in one of the data buffers. This field is usually generated by the MAC chip and the user does not have to account for it in the data buffer. However, in the case of a bridge implementation, the user will normally receive the FCS field and re-transmit the data frame without re-computing the FCS. Only 48-bit addresses are shown here. The user must note that the FSI does not interpret any of the data fields!

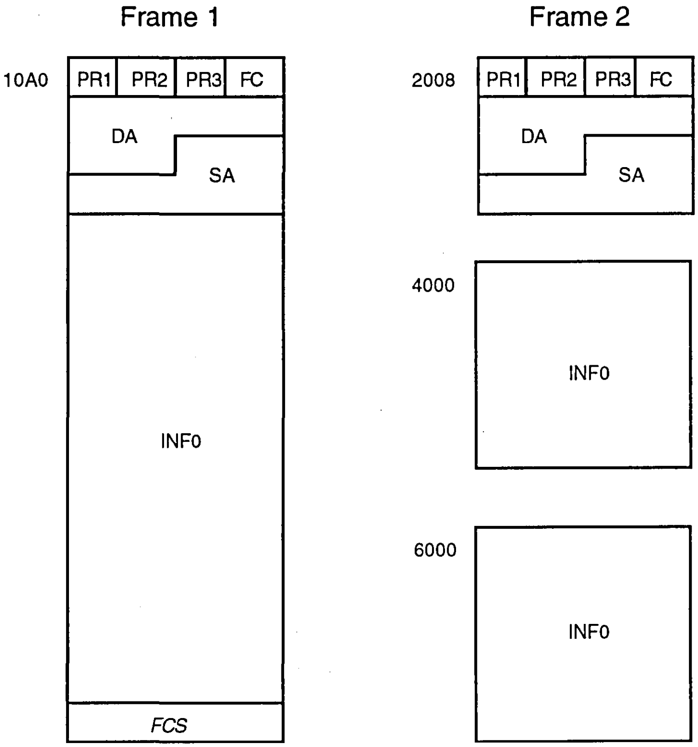


Figure 6-7. Example Memory Data Organization

In the data transmit case the data buffers may be aligned to any byte boundary, furthermore, each data buffer may contain an arbitrary number of data bytes.

In the receive case the data buffers must be aligned to a 32 or 64-bit address depending on whether the user has implemented the 32 or 64-bit bus interface. The length of receive buffer is specified by the RBR register (see Section 3.3.10) or by the Buffer Length Field of the Receive Buffer Descriptor.

Some applications may find the receive data buffer structure insufficiently granular. For example, systems set up to use single receive buffers for maximum length FDDI frames will need to set the receive buffer length to 8K bytes. Note that it is possible to set the receive buffer descriptor pointers to buffers at 4.5K byte increments, but host software should verify that actual frame length did not exceed pointer spacing since the FSI hardware will not check for the user in this case.

6.2.3 Descriptors

The data structure which is used to convey information about the memory data buffers to the FSI are called Descriptors. Descriptors are a kind of command·to the FSI to either transmit, receive or transfer data. Descriptors "describe" each data buffer. There are three types of descriptors:

1. Transmit Buffer Descriptor Command (see section 5.2.3.1)
2. DMA Buffer Descriptor Command (see section 5.2.3.3)
3. Receive or Destination Buffer Descriptor Command (see sections 5.2.3.10 and 5.2.3.5 respectively)

The descriptors contain the address of the data buffer and the size of the data buffer.

Each Descriptor contains an "own bit" to indicate whether the FSI or the host "owns", that is, can use the descriptor and its associated data buffer. Additional control information are the First and Last bits. When a frame exists in one contiguous data block both the First (F) and Last (L) bits are set. Otherwise the First bit indicates the first data block of a frame and the Last bit indicates the last data block.

For data transmission the frame data may be described by up to16 descriptors. A frame which is received, or a data structure which is the destination of a memory to memory transfer, may require the use of an indefinite number of descriptors and buffers. Thus a very large receive frame may be characterized by a large number of small buffers.

6.2.4 Descriptor Rings

The descriptor/commands may be provided to the FSI individually via the FSI Command Register. While this approach may be useful for diagnostics it is inefficient for practical use. To facilitate the FSI's access to the memory data the descriptors may be grouped in rings called Descriptor Rings which are circular arrays in user memory. Defining a descriptor ring is logically equivalent to defining a data DMA channel.

The user can program four Transmit Descriptor Rings and two Receive Descriptor Rings. The Transmit Descriptor Rings are used for queuing frames for transmission, for scheduling commands to be executed by the FSI, and for memory to memory DMA data transfers. The Receive Descriptor Rings contain only buffer descriptors which describe the data buffers for received data. Figure 6-8 shows a small descriptor ring which might well describe the two frames in figure 6-7.

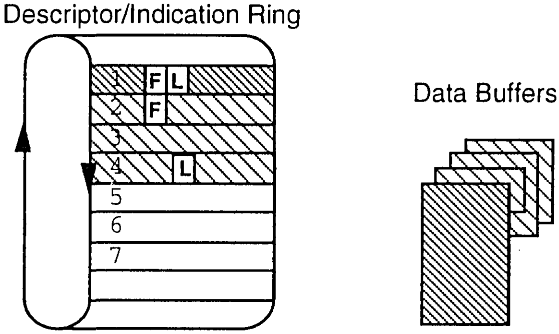


Figure 6-8. Example Descriptor Ring

In Figure 6-8 there are two frames queued. The first frame is contained in one data buffer, indicated by having both the F and L bits set in the descriptor. The second frame is in three buffers and requires three descriptors. The first descriptor has the F bit set, the last descriptor has the L bit set. Not shown is that all the valid descriptors for these frames will also have the Own bit set to indicate to the FSI which descriptors are valid.

The FSI has a Ring Buffer Descriptor FIFO and a Data FIFO allocated in FSI internal memory for each Descriptor Ring. Valid entries in the external Ring Buffer Descriptors are first transferred to the internal Ring Buffer Descriptor FIFO until either the internal Ring Buffer Descriptor FIFO is full (20 entries) or until no other valid external Ring Buffer Descriptors exist. The internal copy of the descriptor Ring reduces bus accesses especially during critical data transfers. This is illustrated in Figure 6-9.

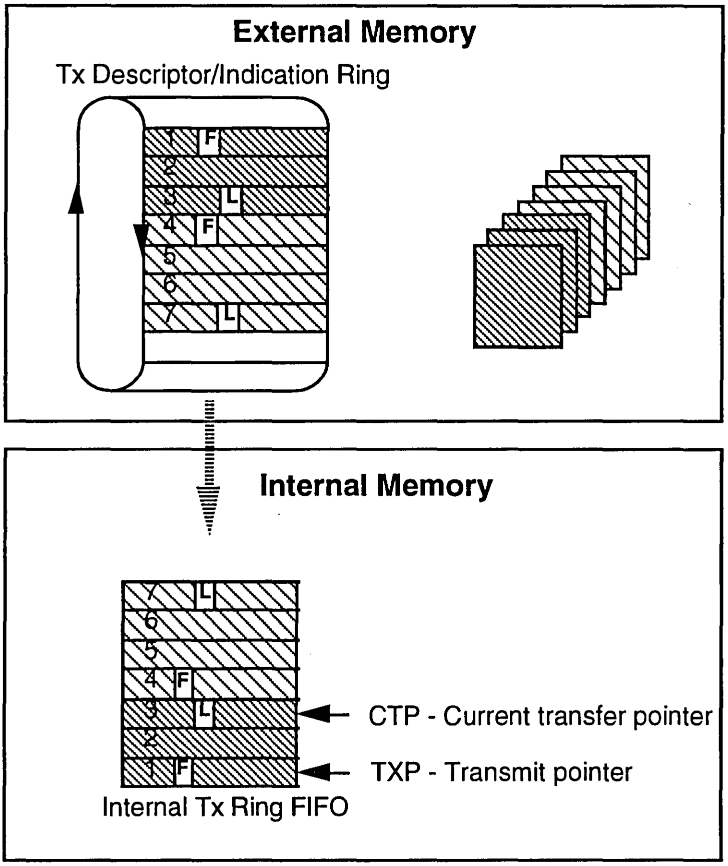


Figure 6-9. Internal Descriptor FIFO

The user must be aware thatdue to the nature of the ring structure an invalid entry, one with the Own bit reset, should be provided following the valid entries to prevent the FSI from wrapping around and re-reading already copied descriptors. To illustrate further, a transmit ring which is designed with an RML of four descriptors and in which all four descriptors are valid will be repeatedly read by the FSI resulting in multiple copies of the ring. The result is that since the FSI operates on its internal snapshot more frames will be sent than desired.

The FSI executes commands sequentially from the internal Transmit Ring Buffer Descriptor FIFO. An internal state machine for each Descriptor Ring handles the pointers to the current command and keeps the state of the Descriptor ring.

The four Transmit Descriptor Rings have a fixed priority between them. Transmit Descriptor Ring 0 always has the highest priority, Transmit Descriptor Ring 1 has the next highest priority, etc. Placement of transmit frames into the proper ring is the responsibility of the host processor and depends on the frame's type and priority. For example, if synchronous frames are to be transmitted properly, they must be placed into Transmit Descriptor Ring 0. Note that, if frames are placed into a higher priority Transmit Descriptor Ring while a lower priority Transmit Descriptor Ring is being serviced, then the frames in the higher priority Transmit Descriptor Ring are serviced as soon as they meet the FSI requirements for transmission.

When observing accesses to a Ring, it is possible for more data and descriptor accesses to occur than might be expected. For example, a Transmit Descriptor Ring containing Transmit Buffer commands followed by a Stop command for the same ring followed by additional Transmit Buffer commands, will likely have some Transmit Buffer commands following the Stop command, along with some of their associated data, transferred to the FSI prior to the Stop command being executed. However, the Stop command is properly executed in order and none of the Transmit Buffer commands following the Stop command are executed until the Ring is restarted.

After the required operation is complete, the FSI writes an indication to the address pointed to by the Ring Write Pointer. This write pointer may point to the same location as the original Buffer Descriptor (writing over the original Buffer Descriptor) or to another location. The indication is written with the Own bit reset.

6.2.5 Destination Rings

Each of the four transmit rings and the two Command channels may have associated destination rings. The Destination Rings define the destination buffers for memory to memory DMA data transfers.

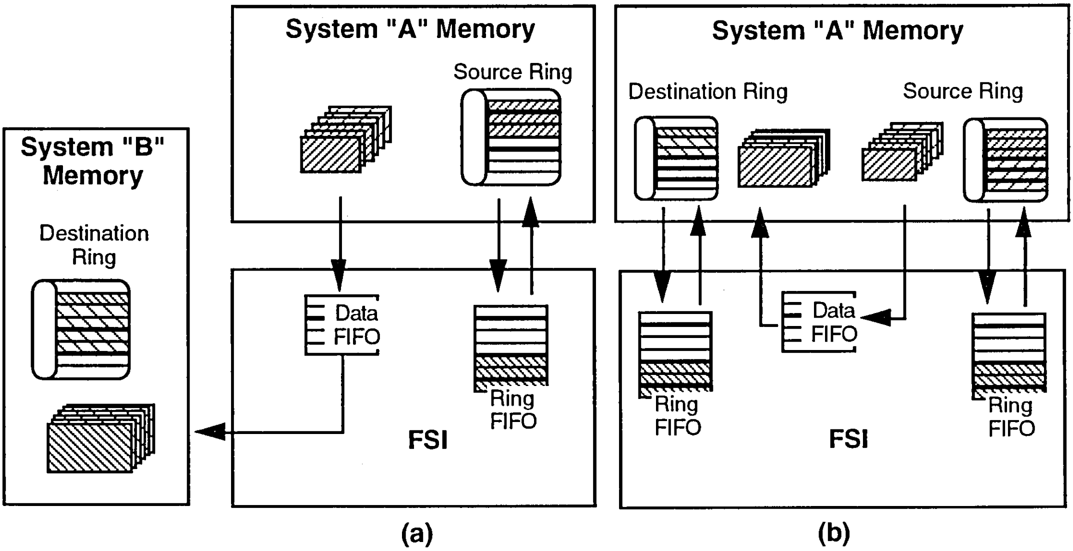


Figure 6-10. Source and Destination Descriptor Rings

To perform memory to memory DMA data transfers a destination ring must be defined by using a Set Destination Ring Command for each destination ring in addition to the definition of the normal, or source, ring.

The DMA data transfer may occur between the memory systems on port A and B or may take place from memory to memory on either port. Figure 6-10a shows a data transfer between ports and figure 6-10b shows the transfer in the memory on one port.

Although Figure 6-10a shows the destination descriptor ring as existing in the same memory as the data buffers it could have been defined and exist in the system "A" memory and only the data accesses will occur over the port to system B's memory. The data buffers at the destination do not have to mirror the source. The DMA data transfer may be used to compact or to scatter data.

6.2.5.1 Buffer Descriptor Ring States

Since data buffer transmission and reception is dependent upon the Buffer Descriptor Ring States, the following is a short description of the Buffer Descriptor Ring State Machine operation.

6.2.5.1.1 State Descriptions

Not Defined The Ring is not defined. No action is taken by the FSI and no internal memory space is reserved for this Ring.

Ready There is at least one entry in this Ring which has not yet been read by the FSI or when the FSI thinks that it can read an entry as when a define ring command is given even though no entries may be valid yet.

Empty The Ring is empty when the FSI decides that all the Ring entries have been read, (i.e., the first entry with the Own bit reset has been read by the FSI). Note, however, that there is at least one entry which has not yet been executed, (either data is still being transmitted or a command is still being executed).

Stopped No other action is taken by the FSI on the entries in the Ring or the data that goes with the entries, even if this data is inside the FSI. In the STOPPED state there are no remaining indications inside the FSI unless the ring stopped itself.

Complete The Ring is empty. All the entries have been executed by the FSI and all the indications have been written back to the Ring.

NOTE

Only when the Ring is in the Complete or Stopped state can the Ring be redefined.

6.2.5.1.2 State Transitions

The following transitions are used in the Ring State Machine shown in the accompanying figure.

A. The Ring is defined and enabled using the Define Ring Command.

B. The Ring is defined by the Define Ring Command but not enabled.

C. All valid entries have been read by the FSI.

D. A Ring Ready control access (access to the FCR register) has been done by the host.

E. All indications have been written back to the Ring.

F. The Ring Stop command has been executed.

G. A Parity or Operational Error has been detected during a descriptor read.

H. The Ring Reset command has been executed.

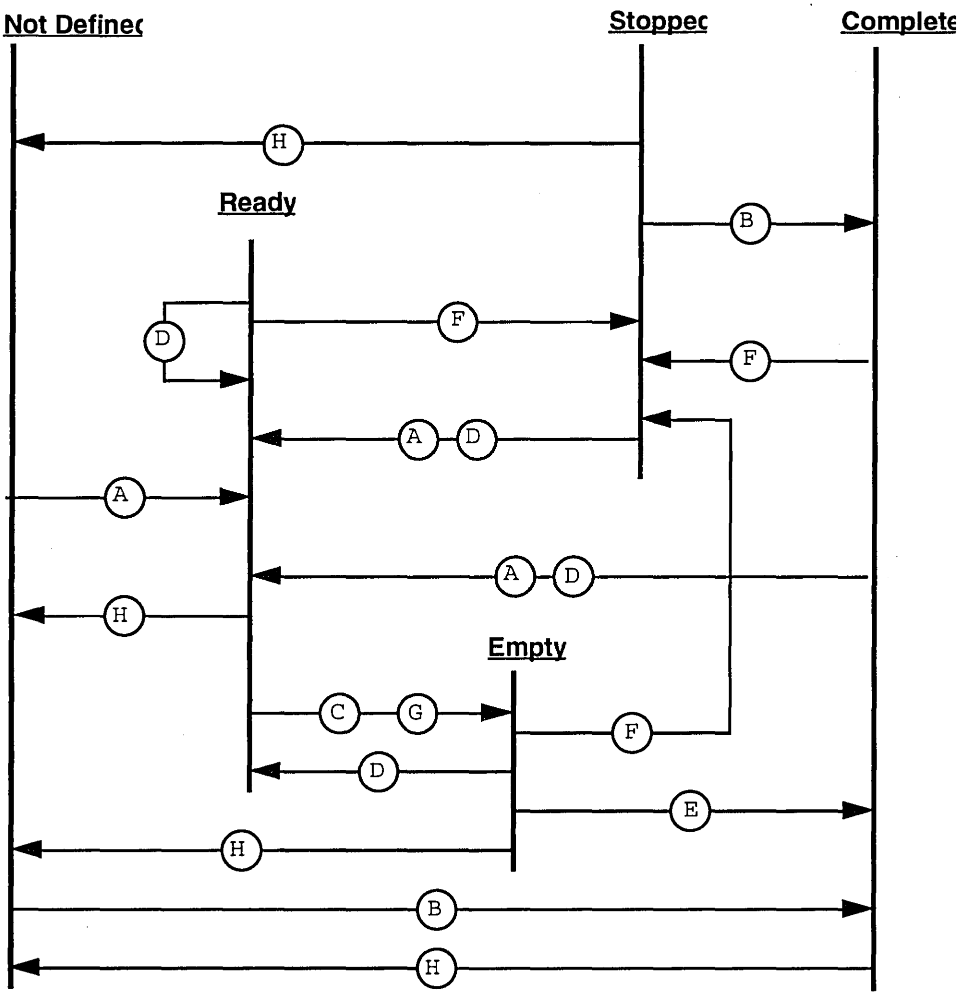


Figure 6-11. Descriptor Ring States

6.3 TRANSMISSION OPERATION

During the transmission process, Buffer Descriptors, and then the data identified by the Buffer Descriptors, are moved into the FSI internal memory. When the appropriate conditions occur: for example, the watermark limit or an end of frame is detected, the frame is then moved into the FSI hardware transmit FIFO and finally presented to the MAC. This progressive movement is controlled by hardware signals from the MAC and the FSI Transmission Logic, the FSI Main Controller, and the FSI Port Control working with the system Bus Control Logic.

6.3.1 Transmit Commands in Rings

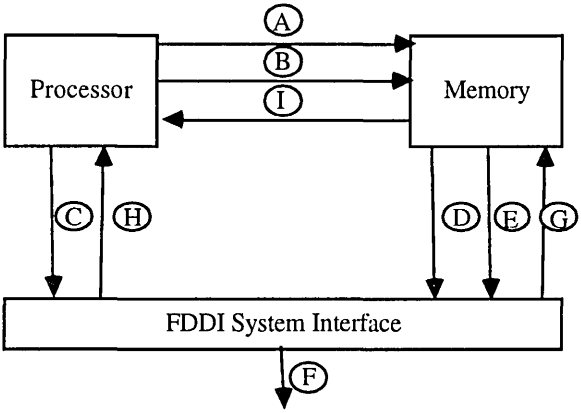


Figure 6-12. Normal Transmission Operation

For a system to transmit a frame using the FDDI System Interface, the following sequence must occur:

A. Data Storage - the storage of data in system memory may happen at any time before the frame is actually transmitted.

B. The host processor prepares Transmit Buffer Descriptors which contain Transmit commands, and writes them into the Transmit Buffer Descriptor Ring.

C. The host processor issues the Ring Ready command.

D. The FDDI System Interface reads the Transmit Buffer Descriptors and the embedded commands and interprets them.

E. The FDDI System Interface DMAs the data pointed to by the Transmit Buffer Descriptors from the system memory into its internal memory to the appropriate watermark.

F. The FDDI System Interface transmits the data onto the network.

G. The FSI generates indications and writes them over the buffer descriptors or to another location as programmed.

H. If enabled, an interrupt is generated.

I. The processor reads the indications.

6.3.2 Linked Ring Transmission

This section illustrates a method of setting up transmission rings to provide for a structured approach to handle multiple frame structures and to reduce the number of interrupts required when handling multiple data units.

In Figure 6-13 Ring 0 is set up to be the central command ring and Ring Command Complete Interrupts are enabled for this ring. Each Define Ring Command is executed in turn when the previous command has been executed. Since each Define Ring Command is synchronized to the state of the target ring, subsequent Define Ring commands for the same target ring will not be executed until the target ring reaches the complete state. The complete state is achieved when the target ring has executed and exhausted all valid entries.

Ring 0 consists of three Define Ring commands. The first two commands point to ring structures for Ring 1 which the FSI will act on. The third Define Ring command points to a null ring structure and is used merely to insure that an interrupt is provided when Ring 1(b) is complete.

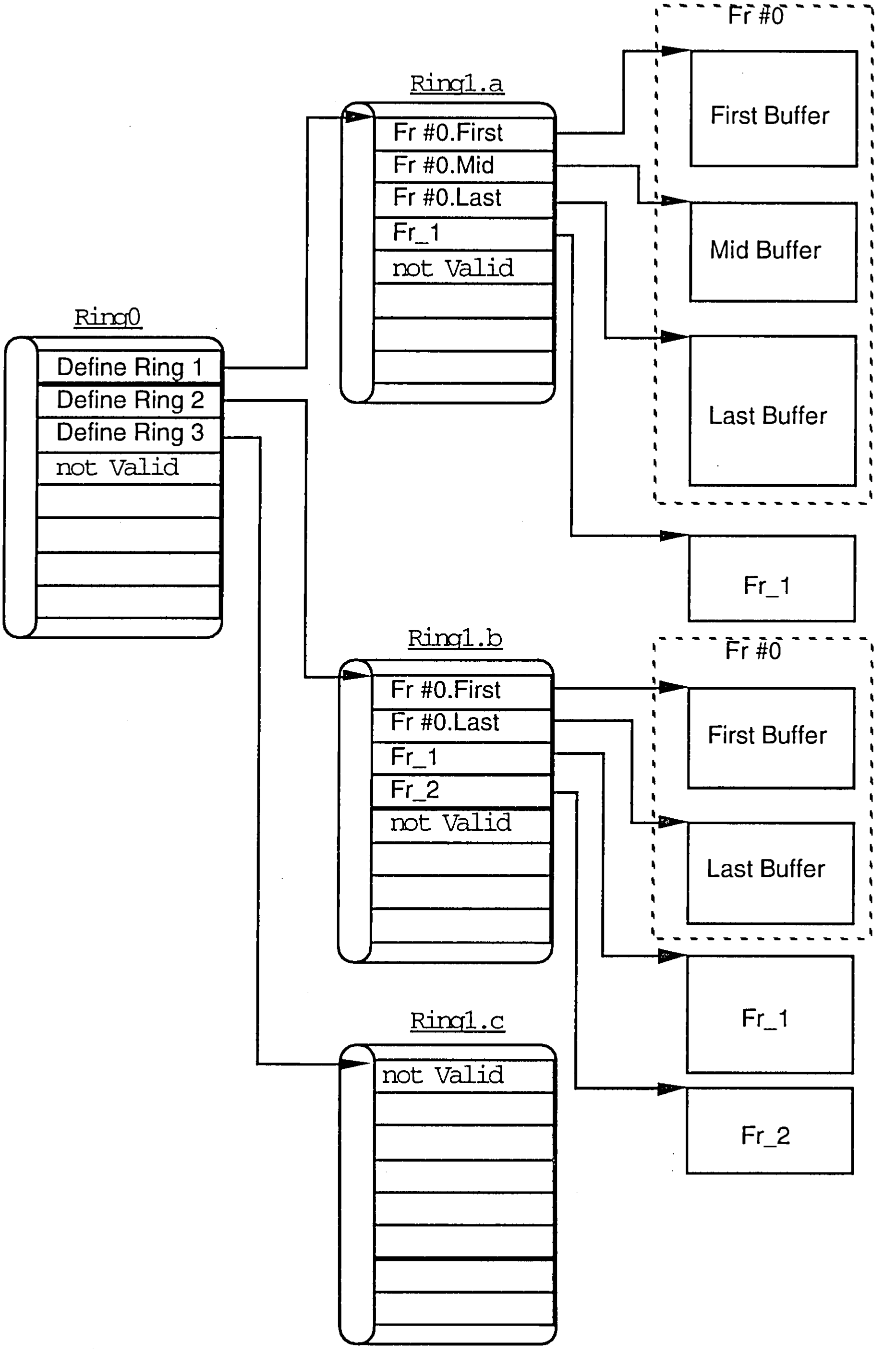


Figure 6-13. Linked Transmission Rings

6.3.3 Endless Transmission

Figure 6-14 illustrates two examples which provide for the endless transmission of frame data by the FSI. In the first example a number of different frames, 4 shown here, can be endlessly transmitted. In the second example a single frame is endlessly transmitted. The latter example may be of use when the FSI is to transmit Directed Beacon frames until informed by SMT to stop such transmissions.

The Ring read pointer used for the Transmit Commands must be different from the Ring write pointer for the indications so that the indications will not invalidate the Transmit Commands.

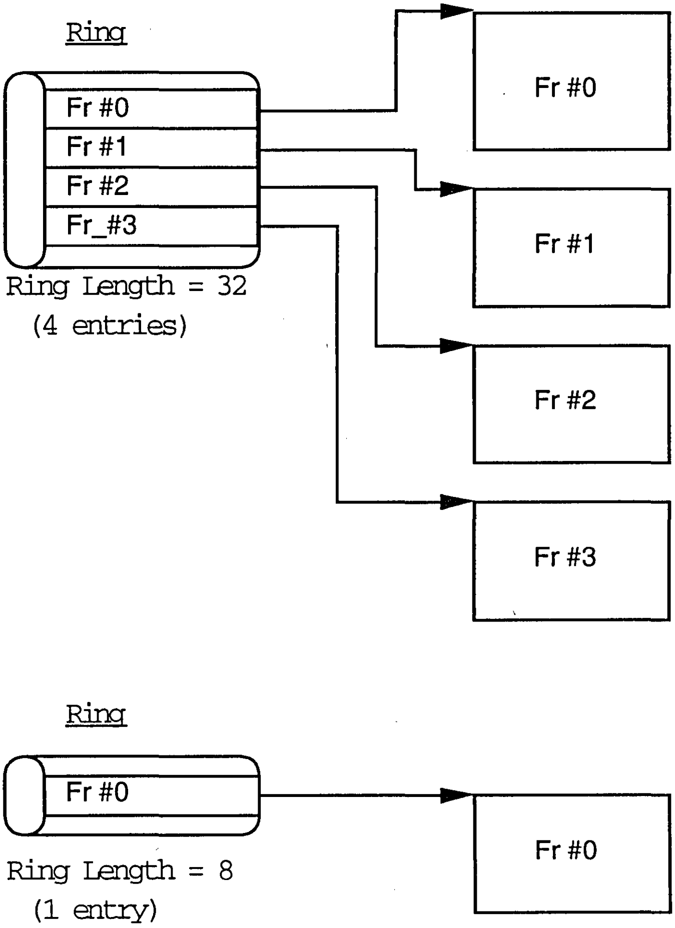


Figure 6-14. Endless Repeat of Frame Data Transmission

6.4. RECEPTION OPERATION

Once the receiver is enabled, the MAC Interface transfers the received data into one of the Internal Receive Data FIFOs according to the FC of the received data frame. When Receive Buffer Descriptors exist (Own bit is set), the FSI reads them from the Receive Ring. If the Receive Ring is defined, the FSI tries to always maintain a number of Buffer Descriptors inside the FSI Internal Ring FIFO in order to reduce the time between reception of a frame into the FSI internal memory, and when the FSI begins to transfer it to external memory. The FSI begins to transfer a frame to external memory when either the full frame exists in the FSI internal memory or the Receive FIFO Watermark has been reached .

6.4.1 Normal Reception — Ring Operation

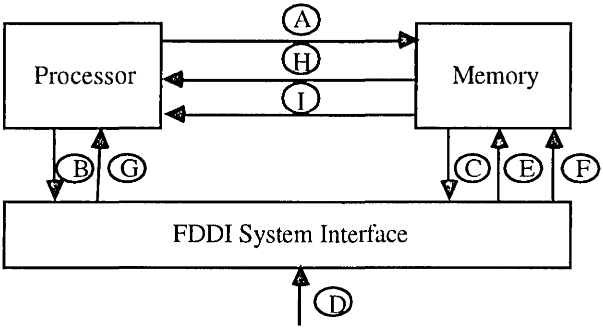


Figure 6-15. Normal Reception by Ring Operation

A. Before a frame can be received from the FDDI network, the Receive Buffer Descriptor Ring must be set up. This is typically done on initialization before the station is active on the FDDI Ring.

B. The processor issues the Ring Ready command to the FDDI System Interface.

C. The FDDI System Interface reads the Receive Descriptor Ring to find the Pointers to the Data Buffers where the FSI will store frames when they are received.

D. A frame starts to be received from the FDDI network.

E. The data is transferred by DMA from the FSI to the external memory.

F. The FSI writes a Receive Indication into all of the Receive Buffer Descriptors of the frame. The information on the status of the entire frame is contained in the last Receive Buffer Descriptor of the frame.

G. If enabled, an interrupt is generated.

H. The processor then reads this Receive Indication.

I. The data buffers may now be read by the processor.

6.4.2 Reception Options

During the reception process, frames are first moved from the MAC into the FSI's hardware FIFO, next into FSI internal memory, and then into the system memory external to the FSI. This progressive movement is controlled by hardware signals from the MAC and the FSI Reception Logic, the FSI Main Controller, the FSI Port Control Logic and the external system Bus Control Logic.

The MAC Interface Logic is responsible for receiving frames and transferring them to the FSI internal memory. During reception, the MAC Interface accumulates small amounts of data in its receive hardware FIFO until the decision is made to start the transfer of this frame into the FSI internal memory. Received frames are checked for their type using the Frame Control (FC) field. The frame may be directed to one of two Receive Data FIFOs according to each frame's FC. This sorting is based on the values specified in the two Receive Frame Type (RFR) registers, which are programmed by the user. If the received frame has its respective type bit cleared inside both of these registers, the frame will not be received and the Receive Abort signal is generated to the MAC to abort reception. The Receive Abort signal is generated in any of the following cases:

a. The FC of the frame being received is not specified to be received in either Receive Data FIFO.

b. The Receive Data FIFO which is selected to receive this frame is disabled.

c. The Receive Data FIFO which is selected to receive this frame is full, in which case an Overrun Error is generated.

d. The REJECT input signal was asserted during frame reception.

The MAC Interface Receiver Logic has the following options of operation:

a. Normal operational mode: receive only good frames that are directed to this station. Bad frames (frames that have been discarded by the MAC), will not normally be transferred to the system. Only when more than one Watermark's worth of information has been transferred into the FSI internal memory will a frame go to the next step, being transferred to the FSI system memory with the appropriate indication in the bad frame's last Buffer Descriptor. If a Receive frame is aborted during reception and the amount of data received is less than one Watermark, then the frame is discarded from the FSI internal memory and its Receive Buffer Descriptor and Data Buffer is reused for another frame.

b. Receive all the data transferred from the MAC to the FSI. This includes partial frames, frames with errors, etc., as defined by MAC programming.

c. Receive One Buffer Mode: this special mode of MAC Interface operation receives only the first buffer of each frame. This mode may be very useful in network analyzers when the MAC is programmed to receive all the frames, (promiscuous mode). In this mode, only a portion of each frame is received starting at the beginning of the frame. This portion is defined by the length of the Receive Data Buffer in the appropriate Ring as specified by the FC of the frame being received. Trailing flags and a complete frame status indication, including the total frame length and the CRC status, are generated by the FSI and put into the frame's indication.

d. Split Header Mode: This mode provides for the extraction and separation of the beginning of a frame from the remainder. The initial portion is called the header and the remainder of the frame is referred to as the data portion. When split mode is used the header portion is always directed to Ring 4 and the data portion is directed according to the setting of the RFR registers.

In addition to the data reception function, the FSI may be programmed to provide an indication of the following events:

a. Frame Sent by this Station. This indication will include flags (E, A and C and any others) as they are received at the end of the frame. This indication of frames that were sent by this station is directed to one of the Receive Descriptor Rings according to each frame's FC field.

b. Token Cycle End. This indication is issued by the MAC when Token, Claim or Beacon frames are received and the FSI has accomplished at least one frame transmission during the current token cycle. The Token Cycle End indication is directed to one of the Rings by the definition of the TE bit in the Receive Frame Type Register (RFR).

The indication for these events is written into a Receive Buffer Descriptor/Indication Ring in one of the two receive Rings. Note that the Receive Data Buffer described by this Receive Buffer Descriptor entry will not be used. The FSI leaves the Buffer Pointer field unchanged to allow reuse of the buffer .

6.5 LOCAL MEMORY OPERATION

The Local Memory may be connected to any port of the FSI. Logically, the Local Memory appears as an expansion of the internal FSI memory and is transparent to the Host Processor. The FSI continues to support all the data structures and the entire set of transmit and receive Rings in the system memory (four transmit rings and two receive rings). The user may choose the Ring mode for each one of the Rings during Ring definition (by using the "Define Ring" command) to be one of the following:

A. Normal Mode - This is the mode of operation when the Local Memory is not used as a temporary storage and frames pass only through the FSI's internal memory FIFOs.

B. Local Memory Mode - This is a mode of operation where Local Memory is used as an expansion to the FSI internal memory.

Some possible data flow configurations with local Memory are shown in Fig 6-16. In this figure two channels are defined to operate in a Normal mode (1,5) and two other channels are operating in a Local Memory Mode (3,4).

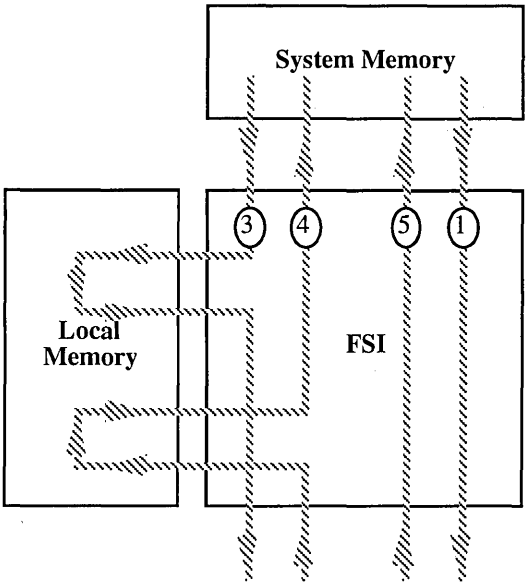


Figure 6-16. Data Flow Configurations

Only in the normal transmit operation are all the commands allowed to be placed inside the transmit (command) rings. These commands will then be executed sequentially. In the Local Memory Transmit mode, only Transmit commands can be placed into the FIFOs. Transmit command execution is complete when the indication has been written back to the ring. For transmission there are two options for the indications. One method is to provide the indication when the frame has been transmitted to the network (or aborted for some reason). The second method allows for the indication to be provided when the frame has been completely transferred to the local memory. In the latter case an indication will not be given when the Frame has been sucessfully or unsuccessfully transmitted to to the network. Frame reception is complete when the frame data and the associated indication(s) have been written to the host memory.

6.5.1 Local Memory Assignment

The Local Memory is considered to be the FSI private memory. Therefore, it should be able to meet the FSI requirements for access. The FSI sees the Local Memory as six large cyclic buffers. Each one of the areas is assigned to one FSI channel that operates in "Local Memory mode"

An example of channel assignment into the Local Memory is shown in Figure 6-17.

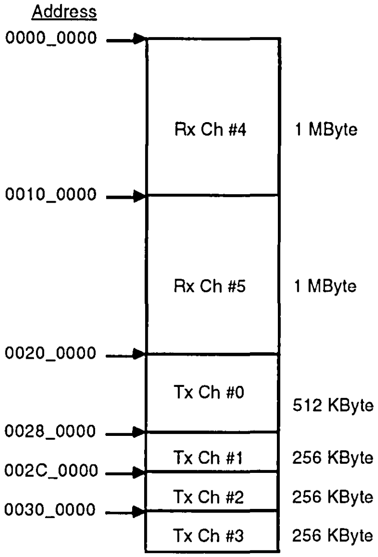


Figure 6-17. Example Local Memory Map

The starting address in local memory for each channel is defined separately using the Set Local Memory Space Command. The channel's assigned area space may vary from 8 KByte to 1 MByte and is specified for each channel by LML parameter in PER (see section 3.3.5). Figure 6-17 shows a method of assigning addresses to each channel's area. The technique assigns the large areas first, starting at the bottom of memory, followed by equal or smaller areas. This assures that successive areas are properly aligned and that the memory space is efficiently used.

In each area space the frames are written one after another without any gaps between them. In the case of multi-buffer transmit frames, the data buffers are assembled and realigned by the FSI such that the frame is stored as a continuous stream of bytes starting on 32-bit word boundaries. The FSI keeps track of how many frames exist inside each area. The maximum number of transmit frames which are waiting to be transmitted inside one Local Memory area is programmable from 1 to 128. When the programmed limit is reached, the FSI will stop transferring frames for this channel from the System memory to the Local Memory. The transfer will continue when the number of frames has decremented below the limit (after a frame has been transmitted).

The maximum number of receive frames which can be waiting inside a receive area is also programmable from 1 to 128 via the Limit Register (LMT). The FSI will stop receiving frames for a Receive channel when the number of frames waiting to be transferred to the system memory in the Local Memory area of this channel reaches the programmed limit. Note, therefore, that an indication for "Local Memory Area Full" is based on the number of frames and not on the actual number of bytes occupied by the frames. Since the FSI treats each memory space for a channel as a cyclical memory buffer, the user can avoid wrapping memory and possibly corrupting data by setting the maximum number of frames multiplied by maximum expected frame size to be less than the assigned memory area for the FIFO in Local Memory.

6.5.2 Transmit Process

As in normal mode, the transmit process starts when the Host Processor has accomplished the preparation of a frame or a number of frames in system memory and informed the FSI that the Transmit Ring is ready (see Figure 6-18). The FSI will then read a number of descriptors from the Ring into its internal memory **(1)** and will start to transfer the data buffers specified by these descriptors from the system memory into the Local Memory **(2),(3)**. This transfer is done in quanta of 256 bytes; i.e., 256 bytes of the frame are transferred from the system memory to the internal FSI memory through Port_In and then from the internal memory to the Local Memory through the Port_Out. Each transfer will take place when its port is available (i.e., not occupied by some other activity). Therefore, one port will not influence the other because of its speed or latency. After the current frame has been transferred to the Local Memory, the FSI will start to transfer the highest priority frame available.

When the entire frame has been transferred to the Local Memory, the FSI generates an Intermediate Transmit Command entry and stores it in its internal memory **(4)**. This entry includes the length of the frame and the start address in Local Memory. It also includes the number of buffers in the original frame and the internal memory address of the last descriptor of the frame in order to put the indication there after frame transmission to the FDDI network. The Intermediate Command entry is used by the FSI in order to transmit the frame from the Local Memory to the network. It is transparent to the user; however, while calculating the amount of internal memory required by the application, the user must take into account that one such entry (8 bytes) will be generated and stored in the FSI internal memory for each frame in the Local Memory. When at least one whole frame exists inside the Local Memory, the FSI will start to transfer the first available frame from the highest priority channel to the internal data FIFO of this channel **(5)** until it satisfies the conditions to transmit this frame (either enough data has been transferred in to reach the Watermark of the FIFO or an entire frame is in the internal data FIFO). The internal Data FIFO treatment algorithm is the same as in normal mode. The user should realize, however, that because of the nature of the Local Memory (which is essentially the FSI's private memory), the Watermarks may be programmed by the user to be minimal (64 bytes). When the frame has been transmitted **(6)**, the indication is generated and placed on top of the last buffer descriptor of the frame **(7)**. The next and last step of the transmit process is to write the indications of the frame to the system memory **(8)**. At this point, all the frame descriptors will be cleared from the internal memory.

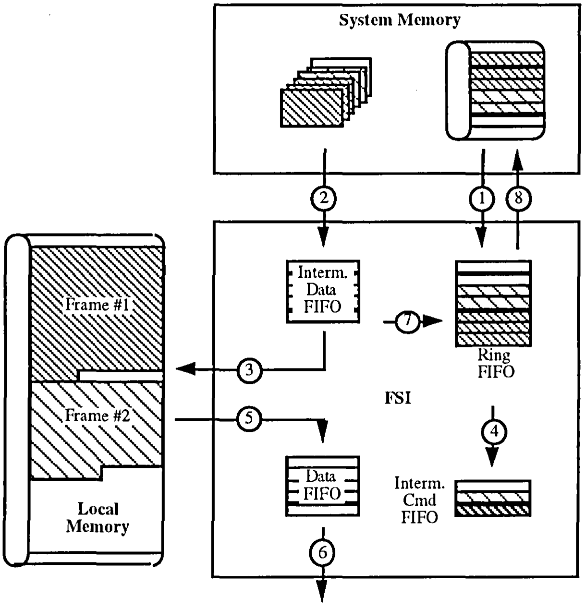


Figure 6-18. Transmit Process

When using the Local Memory operational mode, most of the 8K byte internal memory might be used for descriptors, intermediate commands and a relatively small portion for data because of the small Watermarks required with Local Memory. When indications are to be written after frame transfer to local memory less memory is used than when the indications are held until transmission of the frame.

6.5.3 Receive Process

The Receive Process as shown in Figure 6-19 starts with the reception of data from the MAC into one of the Receive Channels' Internal Data FIFOs **(1)** in the same manner as it's accomplished in the Normal Mode operation. The data will then be written into the Local Memory area assigned to the receive channel **(2)**. When the entire frame has been received, a Frame Descriptor entry is generated by the FSI and stored inside the internal Intermediate Command FIFO of the receive Channel **(3)**.

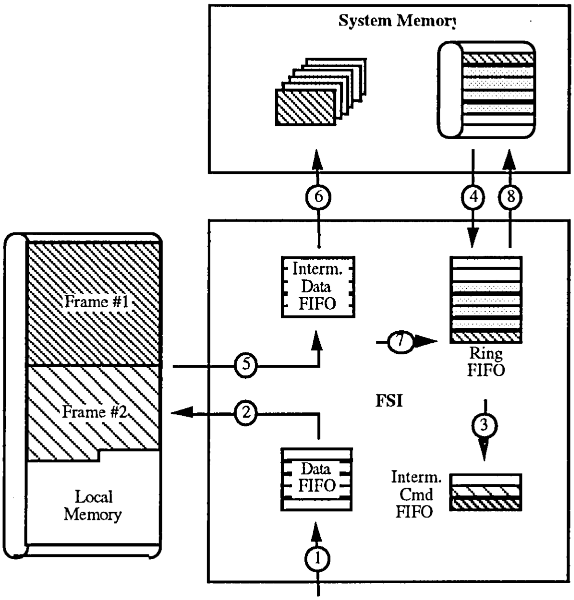


Figure 6-19. Receive Process

When the Ring of free buffers is ready in the system memory, the FSI will read a descriptor from the Ring into the Internal Ring FIFO **(4)**. When a buffer descriptor exists inside the FSI, the Local-to-System transfer is enabled. This transfer is performed through the Intermediate Data FIFO in quanta of 256 bytes **(5)**, **(6)**. When one receive buffer is filled up, the indication is generated by the FSI and placed in the descriptor of this buffer **(7)**. The last step of the receive process is to write the indications of the buffer to the system memory. Note, however, that the internal frame descriptor entry is cleared from the Intermediate Command FIFO after the entire frame has been transferred.

6.6 PORT TO PORT DMA OPERATION

Any Transmit channel not using local memory (up to a total of four channels) and either Command channels (one for each port) may be used in order to transfer data from one system to another (e.g., from a Host processor to a Node processor). It is important to note that DMA operation is not a special operational mode of the channel. Therefore, the same channel may be used both for transmit and DMA operations (see Figure 6-20). However, when mixing Transmit and DMA commands on the same channel the following limitation applies: Either Single Mode should be used for the Transmit commands (bit 45 set) or a Make Indication command should be inserted before each DMA command that follows a Transmit command.

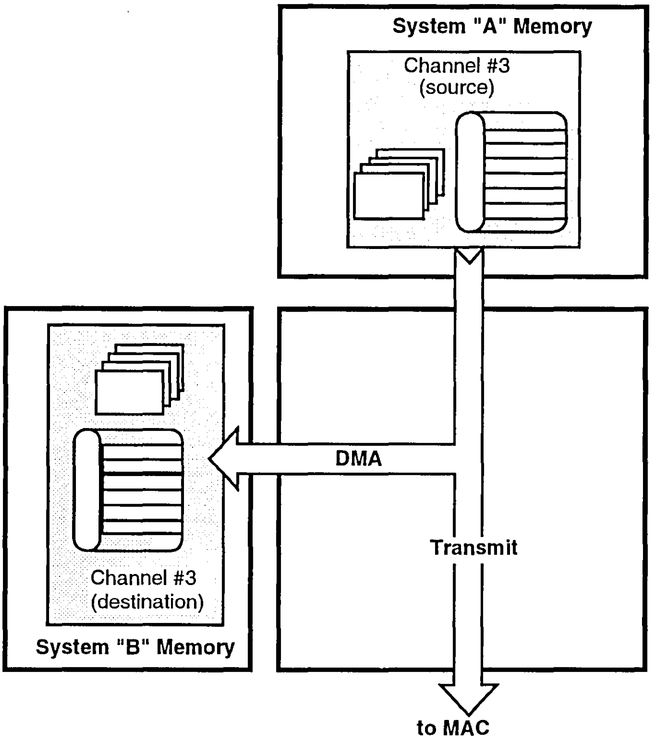


Figure 6-20. Port to Port DMA

An additional property of Transmit and Command Channels, called "Destination Ring," is defined to support DMA operation. The "normal" Ring of a channel will be called the "Source Ring."

The Destination Ring may be defined for every Transmit Channel and every Command Channel and is used to hold descriptors of DMA destination buffers .

The DMA operation is executed by the FSI as a result of a DMA command given through a Source Ring or through the Command Register. The DMA command format is similar to the Transmit Command and may be seen as a transmit operation directed to a destination memory space specified by the Destination Ring descriptors.

6.6.1 Destination Rings

A Destination Ring has the same properties as a Source Ring:

A. A Destination Ring may be assigned to any memory space (A or B).

B. A Destination Ring's data buffers may be placed in any memory space (A or B).

C. A Destination Ring may have its own Ring Maximum Length which may be different from the Source Ring's Ring Maximum Length.

D. Parity may be checked when descriptors are read from the Ring. The Ring states of the Destination Ring are simplified as follows: this Ring may be "Ready" (e.g., there are descriptors inside the ring), or "Not Ready" (e.g., the FSI didn't find a valid descriptor in this Ring). In order to transfer the Destination Ring to the "Ready" state, the "Destination Ring Ready" control access should be performed by the processor (similar to the "Ring Ready" control access for Source Rings).

6.6.2 DMA Operation

DMA operation flow for a Transmit Channel is as follows:

(1) When the Source ring is Ready, a number of descriptors will be read by the FSI into its Ring FIFO (this is the same as in Normal Operation).

(2) When Descriptors exist inside the Ring FIFO, the FSI will execute commands which are specified by these descriptors. If a current descriptor is a first descriptor of the DMA, the FSI will read one destination buffer descriptor from the Destination Ring.

(3),(4) The FSI will transfer the data from source memory space to a destination memory space through the Internal Data FIFO of the channel. This transfer is accomplished in quanta of 256 bytes.

(5) Once one destination buffer has been used, its descriptor is turned into an indication.

(6) This indication will be written back to the Destination Ring. If there is a need for another destination buffer, its descriptor will be read from the Destination Ring and the operation will continue.

(7) When the entire frame has been transferred from the source to the destination, the indication is made and placed inside the last source descriptor.

(8) All the indications waiting inside the Ring FIFO are written back to the Source Ring.

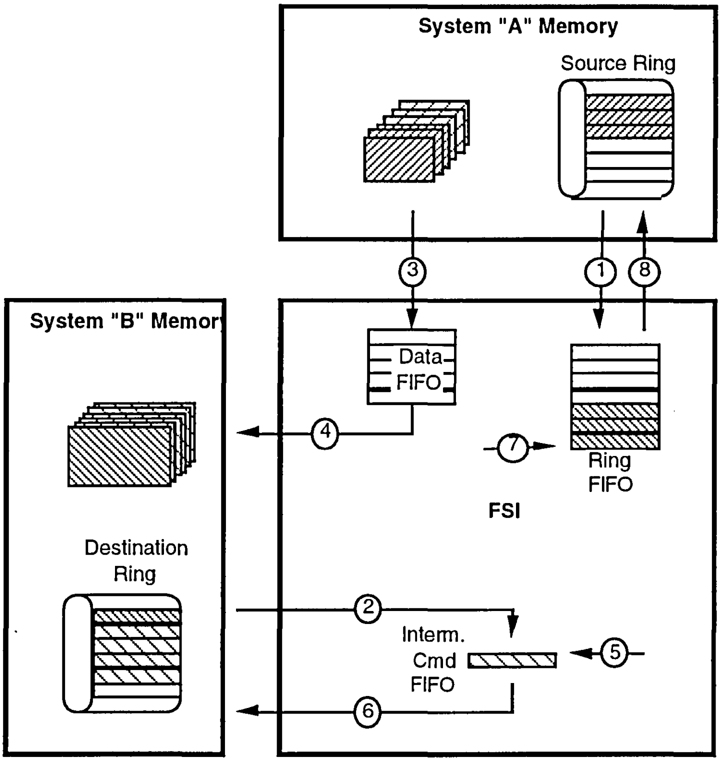


Figure 6-21. DMA Operation for a Transmit Channel

DMA operation using a Command Channel is similar and listed below.

(1) The processor on the source side issues a "DMA" command. As in the operation of the transmit command from the command register, the "DMA" command given through the command register should have only one descriptor; therefore, the maximum amount of data which may be transferred by the command channel by one command is 8K bytes.

(2) The FSI will read one destination buffer descriptor from the Destination Ring of this Command channel.

(3),(4) The FSI will transfer the data from a source memory space to a destination memory space through the Internal Data FIFO of the channel. This transfer is accomplished in quanta of 256 bytes.

(5) Once one destination buffer has been used, its descriptor is turned into an indication.

(6) This indication will be written back to the Destination Ring. If there is a need for another destination buffer, its descriptor will be read from the Destination Ring and the operation will continue.

(7) When the entire frame has been transferred from the source to the destination, the indication is created and placed inside the Command Register.

(8) The "Command Done" status bit in the SR1 is set, causing an interrupt to the source processor (if enabled).

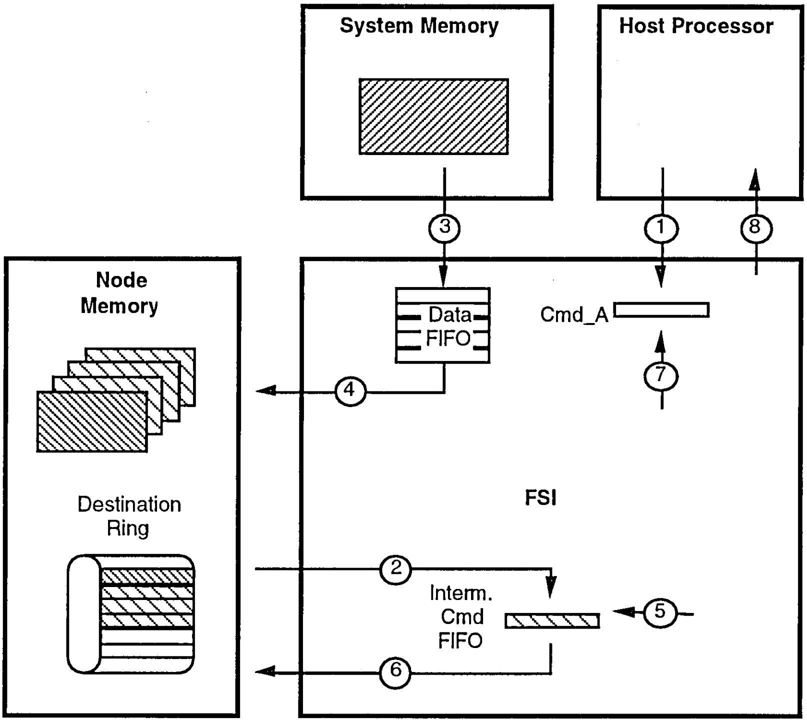


Figure 6-22. DMA Transfers Using Command Register

6.7 HEADER SPLITTING

An additional enhancement to the FSI is the ability to split received frames into two parts (the length of the Header is specified by HLR control register, see section 3.3.11). The first part (the Header) will always be directed to Receive Channel Number 4, regardless of the channel selection decision based on the FC (frame control) and will use an entire receive buffer in this channel . The indication for this buffer will have a unique encoding (S bit set, see section 5.2.3.7) to differentiate this buffer from the normal buffers and will have both "First" and "Last" bits set (therefore the FSI will treat this first buffer as a whole frame). The remainder of the frame is directed to the Receive Channel selected by the

FC of the frame and will take as many buffers as required dependent upon the buffer length programmed for the channel and the remaining frame length. The FSI will treat the second part of the frame as a whole frame as well; i.e., the first buffer occupied by this part will have the "First" bit set in its indication and the last buffer will have the "Last" bit set in the indication. All the indications will have an identifier (S bit set, see section 5.2.3.7) to notify the user that it is the remainder of a frame and not a complete frame.

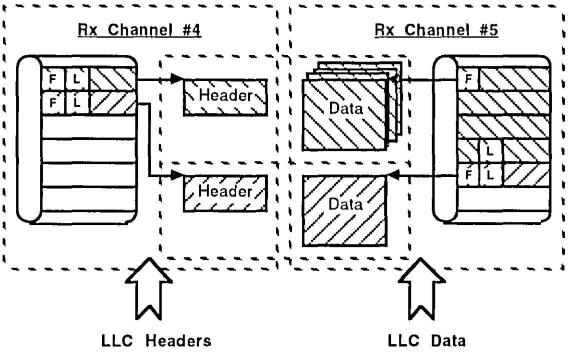


Figure 6-23. Header Splitting

Once the FSI has been programmed to work in the "split" mode, the two sub-modes of operation possible are as follows:

A. The indication for the first buffer (the header buffer) is generated immediately after its reception has been completed.

B. The indication of the header buffer is delayed until the entire frame has been received. Note that when using this mode, the user should program Receive Channel 4 in so that it is used only for Headers.

Due to the possibility that the frame may be aborted during its continued reception after the header buffer has already been received, a Split Mode Data Error Indication may be generated for the second part of the frame.

Because the FSI treats the Header buffer as a small frame, the Receive Complete status bit (RXC(4)) will be set when the indication of this buffer is written back to the Ring. This status bit in the SR1 will cause an interrupt (if enabled) which may actually be considered as an "Early Receive" interrupt when operating in Mode A.

SECTION 7
INITIALIZATION, PROGRAMMING AND EXAMPLES

7.1 INITIALIZATION

After Reset, all internal Control Registers contain zeros, therefore:

A. Port A and Port B are disabled.

B. Transmission from the MAC Interface is disabled.

C. Reception from the MAC Interface is disabled.

D. All RSRs have the EX bit reset; i.e., no rings are defined.

The bits in the Status Register 1 have the following values and meanings:

RNR(5:0)	000000	Rings are not defined yet.
RCC(3:0)	0000	No Ring is Complete.
RXC(5:4)	00	No Receive is Complete.
CRF	1	The Control Register is Free.
CDN	1	Command Done.
RER(5:0)	000000	No Ring Error.
ROV(5:4)	00	No Receive Overrun
POEA	0	No Port A Operation Error.
POEB	0	No Port B Operation Error.
HER	0	No Host Error.
IER	0	No Internal Error.

The bits in the Status Register 2 have the following values and meanings:

INT(7:0)	00000000	No user defined interrupt has occured.
DRE(7:6,3:0)	00,0000	No error has occured on a Destination Ring .
DXC(7:6, 3:0)	00,0000	No Destination Ring is Complete.
DNR(7:6,3:0)	11,1111	No Destination Ring is Ready.

Additionally, the bits of the PSR indicate both Ports are in the Idle state, the Interrupt Mask Registers are cleared which disables all interrupts. All the entries in the internal CAM are undefined and the user must initialize the ENTIRE CAM if the CAM feature is to be used..

Initialization of the FSI should start by programming the internal Control Registers through accesses to the FSI Control Register (FCR):

1. Port Control Registers should be programmed to define the configuration of the external bus connected to each port to enable Port DMA operations. Direct access by the host processor to either Port's registers is always enabled. The PCR for Port A should always be defined. This PCR defines the Parity Control for the entire FSI, including Port B, the internal FSI memory, and the MAC Interface. When Port B is not used as a separate data transfer port, (i.e., it is used for the address associated with Port A data, or in 64-bit mode for 32 bits of the 64-bit data), it should remain disabled. If Port B registers are used, the Host Parity Enable control bit in the PCR of Port B should be defined.
2. Port Memory Page Registers should be programmed according to the External Memory Page used by the system (if any). Note that each Port may be connected to separate memory systems and, therefore, the PMP value may be different for each port.
3. The Ring Parameter Register should be defined for the Ring(s) which are used by the system. Other Ring Parameter Registers may remain undefined and not used.
4. Command Parameter Registers need to be defined if the relevant Command Register is used for issuing Commands directly to the FSI.
5. Parameter Extension Registers need to be defined for any ring using local memory or a destination ring.
6. FIFO Watermark Registers should be defined for all the transmit and receive Rings which are used by the system and for each Command Register that is used for Transmit Commands. See section 7.2.
7. The Limit Registers must be defined for all rings using Local Memory.
8. Receive Frame Type Registers should be defined for each of the Receive Rings if both are to be used. Note that, if each control bit relative to one or several frame types is zero in both RFRs, such frames will not be received and the MAC Interface will generate an RABORT signal to the MAC. If the same control bit is set in both Receive Rings, the frame is directed to Receive Ring #4.
9. The Header Length Register must be defined if operating in Split Header mode.
10. The Maximum Receive Memory Space should be defined for each Receive Ring to be used. Note that the total amount of memory which is used for transmission and reception should not exceed the FSI internal memory size (8K bytes). The FSI will not check the total memory requirements according to the user definitions of the various memory spaces. An Internal Operation Error will occur during operation if the internal memory overruns.
11. The MAC Interface Transmit Control Register should be defined in order to enable data transfer to the MAC. Note that the descriptors and data are transferred by the FSI to internal memory even when the transmitter of the MAC Interface is not enabled.
12. To enable the receive interface the rings to be used are defined and set to ready and then the MAC Interface Receive Control Register should be defined and enabled to specify the operation of the MAC Interface Receiver. Once RE4 or RE5 is set, data may be received and stored inside the internal memory which could lead to a receive overrun if a ring is not prepared to receive the incoming data.

Therefore, the definition of MRR should be the last operation after all Receive Ring Parameters have been specified and the Receive Rings to be used have been defined.

After performing the above operations, the next initialization step is to define all the Transmit Rings which are used by the application. This is done by issuing Define Ring Commands through one of the Command Registers. The CDN status bit in the SR1 should be inspected before issuing each subsequent command. The Define Ring Command may include the Ready Flag in order to transfer the Ring state directly to Ready. If the user is not yet prepared for Ring operation, he may choose to perform a Ring Ready control access at a later time. As soon as one of the Rings is defined as Ready, the FSI will immediately begin Read Descriptor operations.

7.2 INTERNAL MEMORY ALLOCATION

The FSI provides an internal 8K memory space which is used as a latency buffer for data transfers. The internal memory is also used to contain temporary storage for internal snapshots (20 descriptors) of the external descriptor rings and to contain indications before they are written out to external memory.

The internal memory of the FSI is arranged in blocks of 32 bytes each. The internal memory manager requests and frees the blocks as required so that the internal memory is efficiently used during the data transfer operations.

7.2.1 Transmit Ring Allocation

For the purpose of data transmission the user allocates the amount of internal memory used by means of the FIFO Watermark Register (FWR). The Watermark Register should contain a value which is related to the bus latency required to support the working port's bus and prevent transmit underruns from occurring. Deriving this value is explained in section 7.2.4.

During transmit the total internal memory used by the active ring, or channel, is twice the value programmed in the FIFO Watermark Register. This is then the total memory space allocated for the active channel, twice the programmed watermark.

7.2.2 Receive Ring Allocation

In the receive case the memory allocation is set by the Receive Memory Registers (RMR). Again, this value is determined based on the bus latency required to prevent internal memory overruns while receiving data. The watermark setting for a receive ring, or channel, is used to determine how soon the external bus will be accessed for a data transfer, not specifically for bus latency. Since most busses are capable of emptying the internal FIFO very quickly, setting a low watermark means that the external bus will be requested repeatedly following from an emptying/filling cycle.

7.2.3 Watermark Calculation

The following are two examples of the calculation of the maximum internal FSI memory space required for transmission.

Example 1: Three Transmit Rings are defined, each having 256-byte watermarks. The watermark for Transmit Commands issued directly to the FSI is 64-bytes. The calculation is as follows:

256x3 = 768 bytes	The total data space in a Ring non-active state.
256 bytes	Additional data space taken by one of the FIFOs in an active state. In other words an active FIFO can grow to 2X the watermark.
64 bytes	Memory space required by a Transmit Command issued to the FSI through the Command Register. The Internal Command Data FIFO will not occupy any internal space until the Transmit Command is issued through one of the Command Registers. Once such a command is issued, this FIFO may fill up to its specified watermark when not transmitting, and may grow to a maximum of twice its watermark during actual transmission.
160x3 = 480 bytes	The total control space in a Ring non-active state.
128 bytes	Additional control space taken by a FIFO in an active state.
=============	==
1696 bytes	Total FSI internal memory required. Note that the 64 additional bytes required by a Transmit Command issued directly to the FSI will take the place of 256 additional Transmit Data FIFO bytes and are therefore not included in the calculations. Since only one TX FIFO is active at once, if the watermark on the transmit command FIFO exceeded all other transmit FIFO watermarks, then that difference would need to be included. Also, note that any additional space required by the data during transmission from a Buffer Descriptor Ring will not exceed one additional watermark number of bytes as defined for that Buffer Descriptor Ring.

Example 2: Transmit Buffer Descriptor Ring number 0 has a 512-byte watermark. Transmit Buffer Descriptor Ring number 1 has a 64-byte watermark. Transmit Buffer Descriptor Ring number 2 has a 64-byte watermark. Transmit Commands are not issued directly to the FSI through the Command Register. The calculation is as follows:

512 + 64 + 64 = 640 bytes	The total data space in a non-active state.
Max(512,64,64) = 512 bytes	Additional data space in an active state.
160x3 = 480 bytes	The total control space in a non-active state.
128 bytes	Additional control space in an active state.
==============	==============================
1760 bytes	Total FSI internal memory required.

7.2.4 Transmit Watermark Calculations

The calculation of the Transmit Watermark is usually based on the latency of the system bus. The FSI will transfer the data to its internal memory until there is enough data to start the transmit process. Enough data means that there should not be any underrun errors during the transmission. In other words, the data stored inside the internal memory should be sufficient for the period of time between the FSI bus request and when bus grant is received before additional data reaches the internal FIFO. For example, if the bus latency is 50 micro-seconds, the FIFO Watermark should be at least (50000ns./80ns.) = 625 bytes.

7.2.5 Receive Watermark Calculations

The receive Watermark should be defined in order to use the external bus most efficiently. When an external bus is arbitrated by the request/grant mechanism, there is some overhead caused by the arbitration. In order to reduce this overhead per data transfer, the DMA should transfer as many bytes as the system will allow once it gets the bus. The receive Watermark should be defined such that the FSI requests the bus only if it has enough data to transfer. Note, however, that the receive memory maximum space should take into account that all the time of a bus latency the data is received from the network and saved into the internal memory. Therefore, if the bus burst is 128 bytes and the latency is 50 micro seconds, the internal receive data FIFO may grow up to (128 bytes+50000ns./80ns.) =753 bytes.

Again, the rate of transfer from FSI internal memory to the system must approach the rate of frame reception such that internal FSI memory space allocated for reception is not exceeded before the frame has been fully received. Additionally, the average rate of transfer to the system from the FSI must also exceed the rate of reception on the FSI-MAC interface to allow for buffer and descriptor handling overhead.

SECTION 8
PORT OPERATION

The FSI supports two 32-bit ports. The ports are labeled A and B and their respective signals are prefixed with the appropriate port label. The internal FSI Port Control Unit provides address generation, byte swapping and parity checking. With this flexible architecture the user may easily interface the FSI to a wide variety of bus protocols from programmed I/O transfers to full DMA implementations, multiplexed address/data to non-multiplexed address/data, and also generates overlapping data read/write bursts on the same bus tenure. Accesses to Descriptor Rings are not made on the same bus tenure as data accesses.

The FSI is basically a slave device that can become a full bus master with appropriate bus interface timings as provided by a programmable array logic (PAL) type of device. The bus interface logic provides the appropriate control, chip select, and read/write indication to the FSI, and the FSI receives (reads) or provides (writes) the data or provides access to one of the internal registers.

8.1 PORT DATA TRANSFERS

The four types of transfers supported by the FSI are:

Command or Descriptor reads from Command/Descriptor Rings,

Indication writes to Command/Descriptor Rings,

Data Buffer reads of transmit/DMA frames,

Data Buffer writes for received/DMA frames.

The first transfer type is performed by the Port Interface using the Ring Read Pointer (RRP) as defined in the Define Ring command. The "Own" bit, (the most significant bit of the first long word in each command), is sensed during the memory read access and, if it is set (i.e., the FSI is the owner of this entry), the RRP is updated to point to the next entry. If the Own bit is reset, the RRP will not be updated and this ring is Empty. Note that because of the pipeline nature of the ring accesses an extra access is required by the FSI but the data accessed will be ignored. No further accesses for this ring are requested by the FSI until the Ring state becomes Ready again. The RRP is incremented within the limits defined by Ring Maximum Length (RML).

The other transfer types occur based upon the information in the first transfer type, such as receive and transmit buffer pointers, or upon internal information such as current receive or transmit indication pointers.

Parity checking for a Transmit data buffer, if enabled, will occur on the full 32-bits or 64-bits of data on each byte, regardless of actual data byte alignment. The user should ensure that buffer memory has had data with valid parity written to locations that might be accessed by the FSI in such situations.

8.2 PORT SIGNALS

All of the port signals are described in detail in the Signal Description section. The following subsections provide functional descriptions of these signals.

8.2.1 Port Request Signals

The Port Request Signals, AREQ(3:0) and BREQ(3:0), are used to indicate both the internal Port Control Logic state and the next required cycle type as in Read, Write or Idle. Since the FSI is a slave DMA device external logic is used to decode the REQ signals to determine the type of access Required by the FSI.

Each port has three basic states. The following figure shows the possible states through which the Port Control Unit can transition.

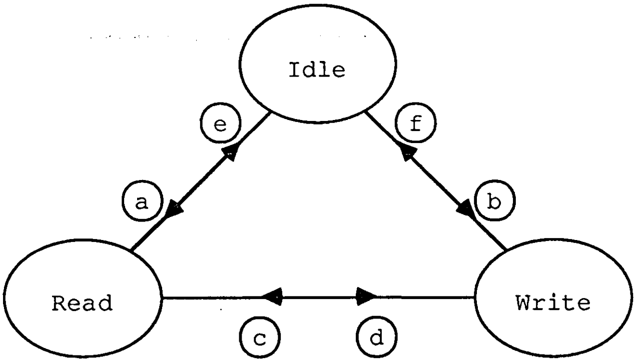


Figure 8-1. Port Control Unit States

Idle: In this state, the port has completed the pending transfers it had to execute. The port will remain in this state until a new transfer is requested from the Main Controller. The port will transition to this state ("e" and "f") when there is no active internal request. This transition is performed during the last data access. In the IDLE state, all requests to the external bus control logic are negated.

Read: The port will transition to this state when a Read operation (Descriptors or Tx DMA read) is required by the Main Controller. Transition "a" from the IDLE state is performed synchronously or asynchronously with the Chip Select signal depending on the mode of port operation.

Write: The port will transition to this state when a Write operation (Indications or Rx DMA write) is required by the Main Controller. Transition "b" from the IDLE state is performed synchronously or asynchronously with the Chip Select input signal depending on the mode of port operation.

The "c" and "d" transitions between the READ and WRITE states take place according to the internal priority when both Read and Write operations are required by the Main Controller. These transitions are always synchronized with Chip Select input signals.

In addition, page information is available to allow efficient use of external dynamic memory. The page output signal on the Request lines (AREQ(0) for port A and BREQ(0) for port B) is asserted when all the following conditions are satisfied:

1. The address of the next data access is on the same memory page with the current data access.
2. The next data access has the same direction (both read or both write) as the current one.
3. The next data access is the same type as the current one (both are Data DMA or Descriptors accesses). If it is a Descriptor access, Read Descriptors or Write Indications, the page signal is negated when the Ring is wrapped.

Since the Request lines are changed to indicate the next required cycle during the access to the Data Register or during a NOP access, the next required cycle type may be sampled by external control logic on the negation of Chip Select. If the system bus is operating in an asynchronous mode, the Request lines will change from Idle to Read or Write asynchronously to Chip Select.

8.2.2 Port CNTL Signals

Internal to the FSI, the CNTL input lines are used to access the FSI. These lines are sampled by the FSI on Chip Select assertion. The CNTL lines access either internal control registers, data address registers, internal data FIFOs, or perform special port functions as with Abort and NOP.

The internal address generator is also updated each data access (access to DTR where CNTL=0010) on Chip Select assertion. If an Address Register (ADR) is accessed, the FSI will latch the value of the address generator on the assertion of Chip Select and this value will appear on the data pins of the selected port. When a multiplexed address/data configuration is used the address of this port is accessed by CNTL=0100. When a non-multiplexed configuration is used and the address appears at a different port from the data, the address is accessed by CNTL=1000 to indicate that the address of the other port is required.

Access to Ports A and B is enabled through the FSI external control pins, ACNTL(3:0) and BCNTL(3:0), for each port when one of the Chip Selects is asserted as defined in figure 7-1. The full address range for the CNTL lines is shown in figure 3-2, only the values directly associated with data transfer are explicated here:

CNTL	Read		Write	
3210	Port A	Port B	Port A	Port B
0000	NOP			
0010	DTR(A)	DTR(B)	DTR(A)	DTR(B)
0100	ADR(A)	ADR(B)	Reserved	Reserved
1000	ADR(B)	ADR(A)	Reserved	Reserved
1110	Abort			

Figure 8-2. FSI Port Direct Access Map

Since the FSI is a slave DMA device the external logic can, at anytime, drive the CNTL and R/W lines to access any of the FSI's internal registers. These accesses can be interleaved with the DMA accesses requested by FSI. That is, the FSI does not have to be in Idle state when the registers are accessed.

The FSI may be wait stated in two ways. In the first approach the chip select assertions of ACSx and BCSx are controlled to assert only when the external logic is ready for the data transfer holding CNTL for either the data or address transfer. The second approach is used if the chip selects are a clock. In this method the CNTL signals must be controlled to provide for the data or address exchange on a single clock cycle and are then NOPed so as to not request new address or data on the following chip select clock.

8.2.2.1 ABORT Access

This access will cause the Port to abort its current operation. It may be used during Transmit data reads, Descriptor reads or Receive data writes and will have the following affect:

A. During transmit data reads, an Abort access will cause the transfer of data into the FSI internal memory to be aborted, potentially making the frame into a fragment. This is similar to the situation that would occur if a parity error was detected during a transmit data read. If the frame does become a fragment, the abort indication is provided in the FSI Transmit Indication.

B. During descriptor reads, an Abort access will cause the FSI to treat the Ring as Empty; i.e., as if it had no additional valid descriptors.

C. During Receive data writes, an Abort access will cause the transfer of the frame being received from the FSI internal memory to abort. The frame data is discarded.

Note that the Abort access has no effect on Indication writes. In 64-bit mode, the abort access should be given to Port A and Port B concurrently. Note that the Abort access has no effect on the request lines of the Port. Therefore, an additional data access is required after the Abort access in order to complete the current data transfer operation.

8.2.2.2 NOP Access

This access has no effect on the internal Port operation and is used only in Port synchronous operation mode to assert the Request Output signals synchronously with the Chip Select input signals. Note that if R/W# is asserted for a NOP access, the FSI will drive the port data bus.

8.3 PORT CHIP SELECT SIGNALS

The FSI provides two chip selects for each port, ACS0, ACS1, BCS0 and BCS1. Only one of the chip select signals may be asserted for each access on each port. Two chip selects per port are provided for the configuration in which more than one master device is operating with the FSI. The chip select signals provide a clocking mechanism for the Port Control Unit.

Asynchronous or Synchronous bus request operation (that is the change from Idle to a New Read or Write state) is selected via bit 5 in each port's Port Control Register (PCR). The function of the mode selection is to determine whether the port state in the REQ signals will change synchronously or asynchronously to chip select. In asynchronous operation the REQ signals will change from IDLE to read or write asynchronously to chip select. In synchronous operation the change from IDLE to read or write will occur synchronously to chip select assertion when the CNTL input lines are performing a NOP access. After the read or write cycle has started, that is, the interface goes from IDLE to read or write, subsequent state changes are synchronous to chip select.

8.4 PORT OPERATION

The external processor(s) and external Bus Control Logic will normally operate on their own clock as defined by the external bus timing. All transfers to and from the FSI are synchronized inside the chip; therefore, no external synchronization is required. The external timing is defined by ACS0 and ACS1 for Port A, and by BCS0 and BCS1 for Port B. All Port bus activities are defined with reference to these select signals. Note that Port A may use an external clock which is different from Port B.

If asynchronous port operation has been selected in the Port Control Register, it should be noted that the Bus Control logic has to synchronize the transition of request lines from the Idle state to the Active state in conjunction with the system clock in order to correctly recognize the next required cycle. The chip select signals are used to acknowledge readiness for the external logic to perform the required cycle.

If synchronous mode of port operation is selected in the Port Control Register the Request Output Lines of each port are changed only during the active cycle of one of the Chip Selects. The Chip Select signals are the clock signals for synchronous operation and all timing is relative to these signals. Therefore, if the port's current state is Idle, the transition to the Active state will occur synchronously with the Chip Select line and may be sampled by the Bus Control Logic when the Chip Select line is negated. Note that this transition will take place only when the external Bus Control Logic performs a NOP access to the port.

8.4.1 Port Address Generation

Each transfer of data or command/indication information has its address generated internally by the FSI Port Control Unit. This address is updated each time the data transfer is performed. The next address to be accessed is compared with the current access address in order to decide if the next access is on the same memory page as the current access. The address may be accessed at any time between data accesses.

8.4.2 Interport Operation

When used as two separate, 32-bit ports, the FSI offers the system designer the flexibility to provide capability such as FIFO descriptors and indications on one port and FIFO data on a separate port. In these modes, the following considerations should be taken into account:

A. It is possible to interrupt a processor on Port B, for example, with events happening on Port A using the General Status Register (SR1). This is possible since the SR1 is common to both ports and accesses to this register will show the same information except for Command Done (CDN), Control Register Free (CRF), and Host Error (HER) bits which reflect status for each of the ports individually. As an example, a Ring which transitions from the Ready state for a Ring in Port B's memory space may cause an interrupt to the Port A processor if the RNE bit for this Ring is set in the Interrupt Mask Register (IMR1) of Port A.

B. Direct access registers, such as the Interrupt Mask Register (IMR), Command Register (CMR), Command Extension Register (CER), and Port Status Register (PSR), etc. are divided into Port A and Port B registers and cannot be accessed from the other port.

C. All internal parameters, including the Port control and Ring parameters for each of the Rings, are accessible by accessing the FSI Control Register and may be set from either or both Ports.

8.4.3 Port Operational Errors

The Port Operational Error (POE) condition occurs when a system bus error is detected by bus control logic. In such a situation as, for example, when the bus control logic requests a data access and the data access acknowledgment has not been received by the bus control logic quickly enough, POE can provide an interrupt to the host processor. In bus error situations, the bus control logic needs to abort the current data access and create an abort. In order to accomplish this, the bus control logic should assert the second chip select on the offending port causing both chip selects to be active at the same time. In response to this abnormal operation, the FSI will set the POE status bit and reset the Port Enable control bit. The FSI will ignore the new request it has generated and will transition to an Idle indication on the request lines a delay time after the second chip select has been asserted. Depending upon the current transfer type, the effect is:

a. For transmit or receive descriptor reads, the Ring state is changed to Empty with the Operation Error status bit set in the appropriate Ring State Register. The RNR and RER bits in the SR1 are set.

b. For transmit data reads, setting the POE bit will cause transfer of a transmit frame into the FSI to abort, potentially making the frame into a fragment. This is similar to the situation encountered if a parity error is detected during transmit frame data reads. The indication generated by the FSI will contain the abort indication if it occurs.

c. When the FSI is performing Receive data writes, setting the POE bit will cause transfer of a received frame to the memory to abort. The indication generated by the FSI will contain the abort indication

d. Setting the POE during Indication Write operation has no effect on the operation.

8.5 PROGRAMMED I/O OPERATION

The FDDI System Interface can operate in a programmed I/O mode. In this method of interfacing to a system the data is transferred by either the high speed processor itself or by another high performance peripheral in the system.

The FSI itself does not perform any bus control functions and is totally indifferent to how, and by what device, bus control cycles are implemented. Therefore, the host processor may actually handle all transfers without any Bus Control Logic. In this case, the host processor may choose to give commands to the FSI through a command/descriptor ring or through the command and command extension registers. These methods are described in the following sections. In order to perform transmission by issuing commands through the command/descriptor rings, the host processor performs the following procedure:

a. The host sets the Ring to Ready using a write operation to the Control register. As a result, the FSI will transition this Ring state to Ready and will request a Read Descriptor cycle.

b. Then the host writes the Transmit Buffer Descriptor entry into the FSI when it senses the port request, either by reading the Port Status Register or by getting an interrupt based on the external Request Signal inputs. Using these write cycles, the host processor may enter a number of descriptors inside the internal Transmit Ring FIFO. To stop the Read Descriptor cycle of the FSI, the host processor writes the last entry with the Own bit reset. When the Port Interface reads this entry and determines that is doesn't belong to the FSI, it stops the Read Descriptor cycle and this Ring transitions to Empty. The next FSI cycle is the Read Data cycle to fill the Internal Transmit Data FIFO up to its Watermark.

c. The host writes the data into the FSI when it senses the port request. When the internal Transmit Data FIFO reaches its Watermark, the port request is negated. The Host processor may use the Idle state condition on the Request signals in order to get an interrupt. The Read Data cycle request is asserted again by the port when the FSI's internal FIFO starts to transmit the data to the MAC. The host processor should then continue to write the data into the FSI. After the frame is transmitted, the port initiates a Write Indication cycle.

d. Finally the host reads the indications from the FSI when it senses an appropriate port request. As in the case of data, the port may go to the Idle state after a number of indications. Additional attempts to read the indications are ignored by the FSI. When the last indication of a frame is transferred to the host, the Transmit Complete status bit in the SR1 is asserted.

Note that the user should not perform data accesses to the FSI when the FSI is in Idle state, i.e., when the REQ lines indicate Idle.

8.5.1 Programmed I/O Transmission

In order to perform slave transmission using commands given directly to the FSI, the host processor performs the following procedure:

a. Write the Transmit Buffer Descriptor into the FSI Command register. When this command is written, the FSI executes it by trying to get the data for this frame. Therefore, the port initiates a Read Data cycle.

b. Write the data into the FSI after it senses the port request. When the internal FIFO has reached its Watermark, the port request is negated. The host processor may use the Idle state condition on the external Request signal lines in order to get an interrupt. All the additional data write accesses which are executed by the host processor after the port Idle state is reached are ignored by the FSI and the host processor should repeat them after handling the interrupt. The host processor may read the internal FSI address or transfer counter to determine what data it should repeat. The Read Data cycle request is reasserted by the FSI when the FSI's internal FIFO starts to transmit the data to the MAC. The Host processor should then continue to write data into the FSI. After the frame has been transmitted, the FSI writes the indication inside the Command Register, resetting the Own bit of the command. This may cause assertion of the Command Done interrupt if it was previously enabled. Note that the Command Register remains occupied until the completion of transmission. Therefore, no other command can be issued from this command register during this period.

Transmission using Transmit Commands issued directly to the FSI may be used in order to meet the FDDI standard's requirement for immediate frame transmission. Such frames must be able to supersede all other frames queued for transmission. This mechanism may also be used to transmit other types of frames in addition to immediate frames. These frames may be of high priority or frames that do not properly belong in one of the four transmission rings.

8.6 FUNCTIONAL PORT OPERATION EXAMPLES

The following figures illustrate some functional timing examples for the FDDI System Interface.

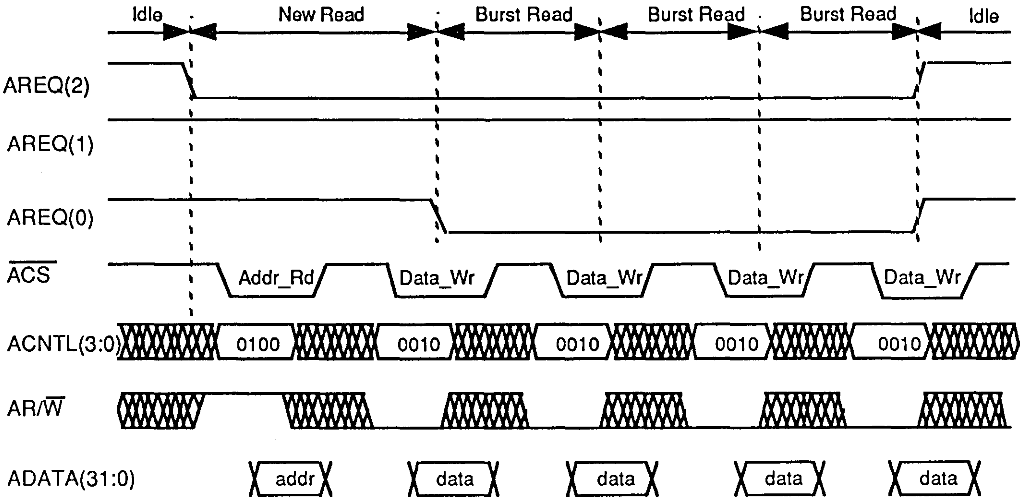


Figure 8-3. Address/Data Multiplexed Operation

This figure shows four accesses of data from the same external memory page. A preceding access to the Address Register is shown to determine the start of this burst. In this figure as well as in the next two figures, the ambiguous time frames for the control and R/W lines as indicated around the rising edge of chip select, are meant to emphasize that the FSI samples the Control and R/W lines on Chip Select Assertion.

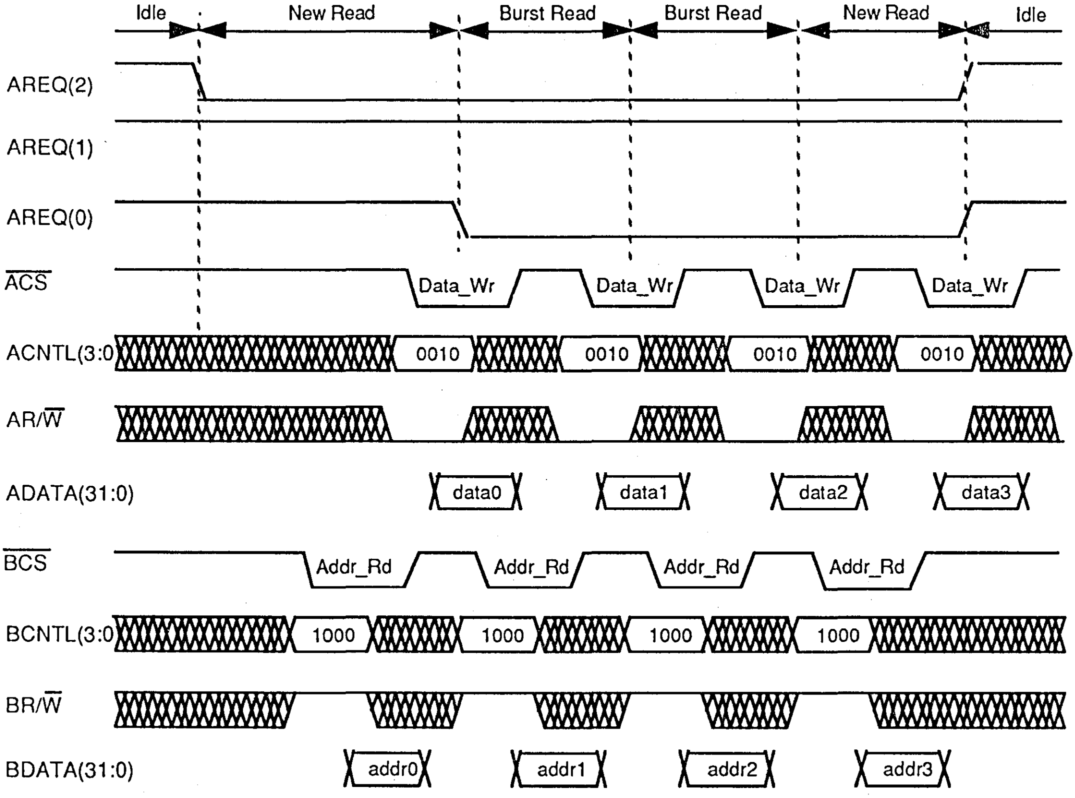


Figure 8-4. Non-Multiplexed Address/Data Operation

In figure 7-4, Port A is used for 32-bit data and Port B is used for 32-bit address. Note that the address read is done prior to the data access in order to reduce the cycle time. The address is changed on the falling edge of the data access and it may be read out on the following address access.

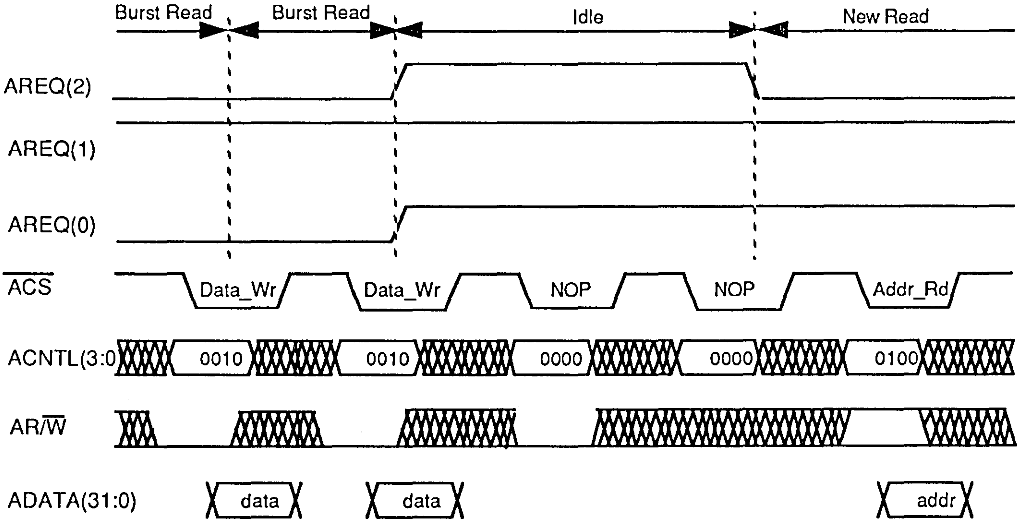


Figure 8-5. Synchronous Operation

Figure 8-5 demonstrates Synchronous Port Operation, the Request lines are shown as change from Idle to New Read when ACSx is asserted and the ACNTL = NOP.

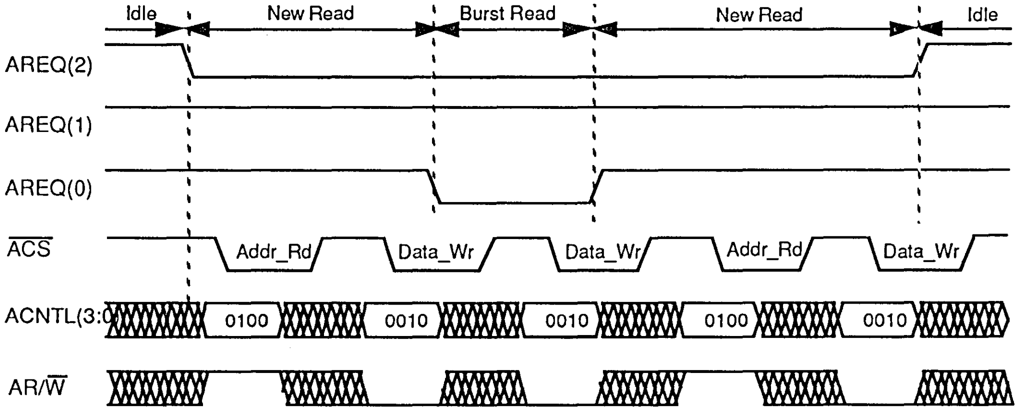


Figure 8-6. Data Read Operation With a Change of Page

Figure 8-6 shows an asynchronous bus cycle, that is the change of state from Idle to New Read is done asynchronous with respect to chip select, ACSx. In this case the bus logic would have had to synchronize the change of state on the REQ lines and then present ACSx when ready to proceed with the remainder of the bus tenure. The initial access is a New Read in which an address is presented followed by a single data transfer for that address, the Burst Read, and then another New Read cycle due to a change of page.

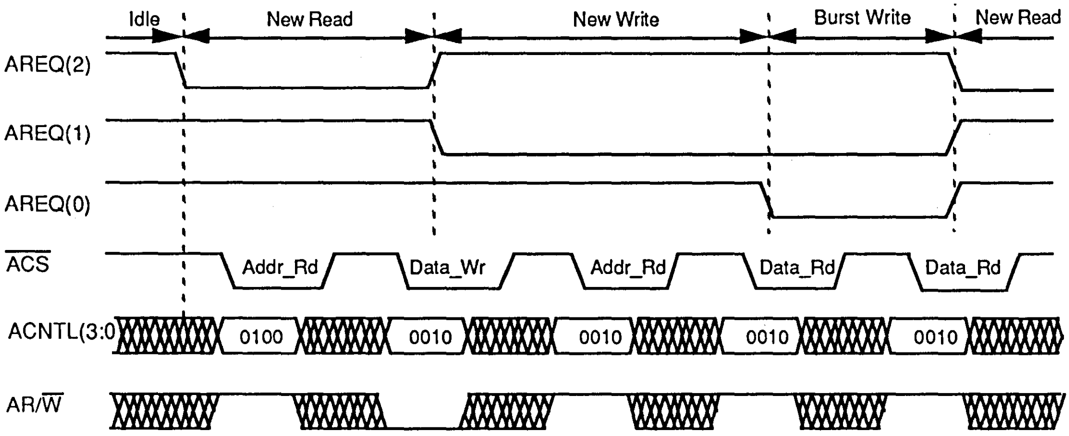


Figure 8-7. Data Read Followed by a Data Write in the Same Bus Tenure

Figure 8-7 illustrates the interleaving of read and writes on a single bus tenure. Figure 8-7 shows an asynchronous bus tenure in which a single read access is followed by a write access followed by a read access.

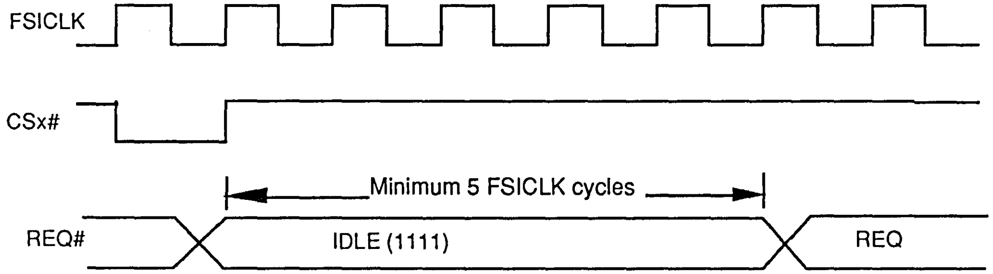


Figure 8-8. Request Negation to Request Assertion in Asynchronous Mode

SECTION 9
BOUNDARY SCAN

The FSI implements standard test access port and boundary scan capabilities as per the IEEE P1149.1, standard also known as JTAG. This facility allows for the isolation of the FSI for device testing and for board testing.

9.1 JTAG OVERVIEW

The test problem for any product constructed from a collection of components can be decomposed into three goals:

1. To confirm that the components are interconnected in the correct manner.
2. To confirm that each component performs its required function.
3. To confirm that the components in the product interact correctly and that the product performs its intended function.

The first two goals can be achieved through the use of a Boundary Scan Register architecture. The third goal must be achieved either using a functional ATE system or using a system level self-test, in this case with appropriate loop-backs, etc.

The boundary scan technique involves the inclusion of a Shift Register stage adjacent to each component pin. These Shift Register Stages, called Boundary Scan Cells, are interconnected to form a shift register chain around the FSI Chip I/O pins. The shift path is provided with a serial input (TDI), a serial output (TDO) and an appropriate clock (TCK). The control of the shift action, the parallel output of the shift register to the output pins, or the parallel output of the shift register to the interior of the chip is achieved by an appropriate sequence of logical values on the TMS signal.

The JTAG technique of testing implemented in the FSI can detect many of the faults that testers currently address without the need for extensive bed-of-nails access and without the need of expensive test equipment, particularly for surface mount components.

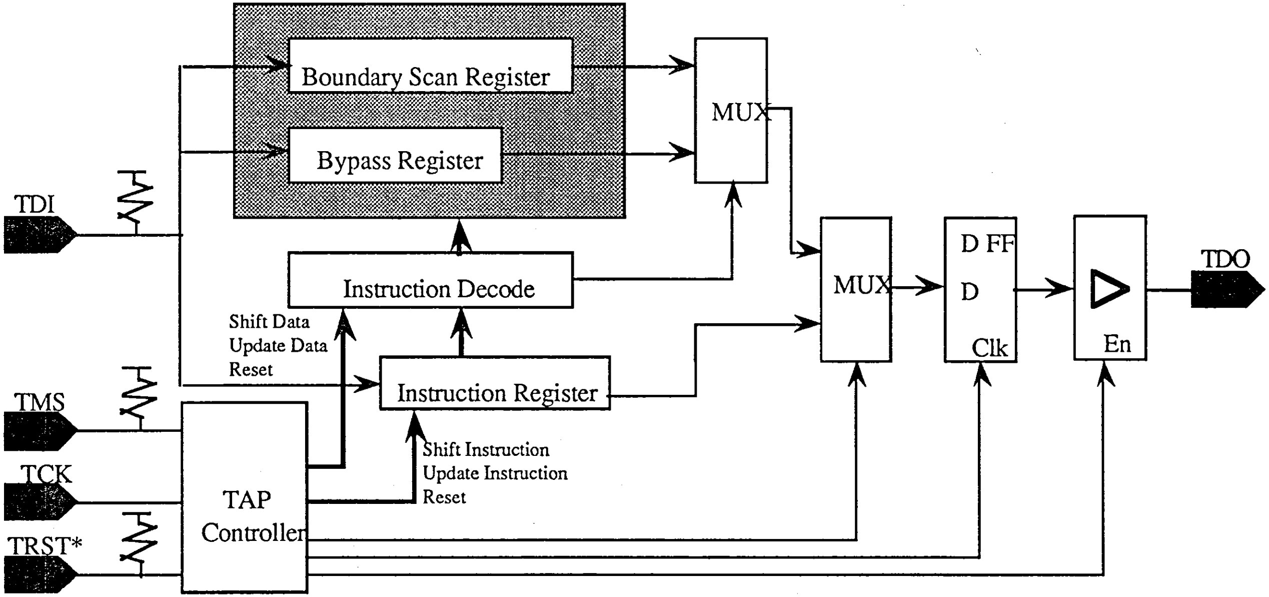


Figure 9-1. JTAG Architecture Model

9.2 FUNCTIONAL BLOCKS

9.2.1 Test Access Port (TAP) Controller

The TAP Controller is responsible for interpreting the sequence of logical values on the TMS signal. It is a synchronous state machine which controls the operations of the JTAG Block. The state machine is shown in the following figure where the value shown adjacent to each arc represents the value of the TMS signal sampled at the rising edge of the TCK.

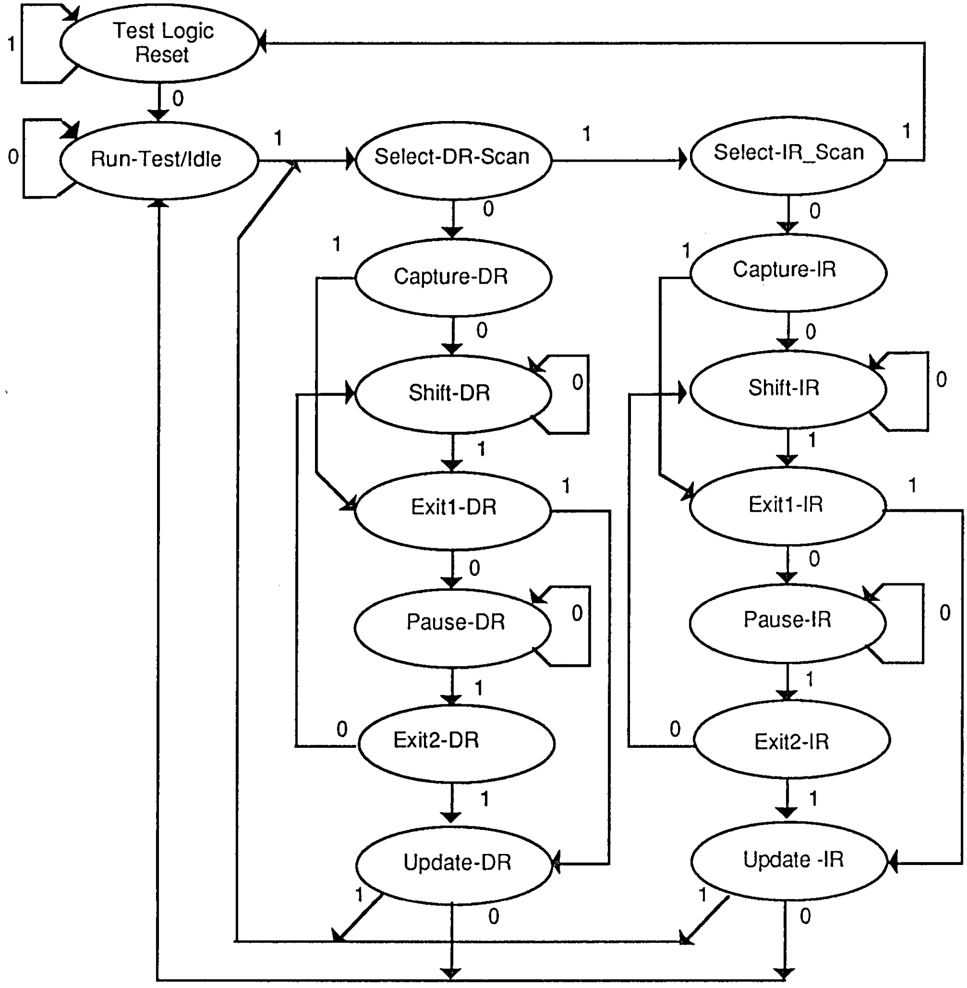


Figure 9-2. TAP Controller State Machine

9.2.2 Instruction Register

The Instruction Register is used to select the test to be performed and/or the test data register to be selected. It is serially loaded in the Shift-IR state. The data loaded is transferred to the parallel output and interpreted as an instruction when the TAP is in the Update-IR state. When the TAP is in the Test Logic Reset state, the Instruction Register is set to select the Bypass Register.

The Instruction Register is a 2-bit register, I(0) and I(1). I(0) is the first bit to be output to the TDO when the TAP is in the Shift-IR state.

When the TAP is in the Capture-IR state, the Instruction Register is loaded in parallel with the logical value I(0)=1, I(1)=0.

The Instruction code is given to the Instruction Register with the rightmost bit being the closest to the TDO output:

11	BYPASS Instruction
10	SAMPLE/PRELOAD Instruction
0x	EXTEST Instruction

9.2.3 Boundary Scan Register

The Boundary Scan Register can be viewed as a parallel-in, parallel-out shift-register with some additional functionality; e.g., the ability to propagate data directly from the parallel input to the parallel output without clocking.

The Boundary Scan Register allows testing of circuitry external to the FSI. It also permits the signals flowing through the system pins to be sampled and examined without interfering with the operation of the FSI.

The Boundary Scan Register includes all the pins of the FSI (except the 5 pins that belong to the JTAG interface), and 4 additional cells that determine the direction and driving state of the ADATA(31:0), APRTY(3:0), BDATA(31:0) and BPRTY(3:0) busses.

9.2.4 Bypass Register

The Bypass Register provides a "short circuit" route for test data in the data register scanning cycle. Use of the Bypass Register can speed access to, for example, test data registers in other components on a board level test data path.

The Bypass Register consists of a single shift register stage that is connected between TDI and TDO when the BYPASS instruction is selected.

When the TAP is in the Capture-DR state the Bypass Register is loaded with zero.

9.3 JTAG INSTRUCTION SUPPORT

The FSI supports the BYPASS, SAMPLE/PRELOAD, and EXTEST instructions:

BYPASS The operation of the JTAG Block has no effect on the operation of the FSI. The BYPASS Register is selected to be connected for serial access between TDI and TDO during the appropriate state machine states.

SAMPLE/PRELOAD The Boundary Scan Register is selected to be connected between TDI and TDO. The FSI continues its normal operation and the signals entering and leaving the FSI are sampled without affecting circuit operation.

EXTEST The Boundary Scan Register is selected to be connected between TDI and TDO. The cells at Output Pins are used to apply test stimuli while those at Input Pins capture test results. In the Capture-DR state, all the signals received at the input pins are loaded into the Boundary Scan Register.

9.4 BOUNDARY SCAN CONTROL

The following signals are internal control signals which are included in the scan chain; their function is as noted below:

A_IEN is the Direction Control of ADATA(31:0) and APRTY(3:0) lines for the EXTEST instruction where:

A_IEN = 1 The I/O Buffer of the lines mentioned above are configured as inputs.

A_IEN = 0 The I/O Buffers of the lines mentioned above are configured as outputs.

A_OEN is the Active Control of the ADATA(31:0) and APRTY(3:0) lines for the EXTEST instruction where:

A_OEN = 1 The Output Buffers of the lines mentioned above are in Active state.

A_OEN = 0 The Output Buffers of the lines mentioned above are in Three-State.

B_IN is the Direction Control of the BDATA(31:0) and BPRTY(3:0) lines for the EXTEST instruction where:

B_IEN = 1 The I/O Buffer of the lines mentioned above are configured as inputs.

B_IEN = 0 The I/O Buffers of the lines mentioned above are configured as outputs.

B_OEN is the Active Control of the BDATA(31:0) and BPRTY(3:0) lines for the EXTEST instruction where:

B_OEN = 1 The Output Buffers of the lines mentioned above are in Active state.

B_OEN = 0 The Output Buffers of the lines mentioned above are in Three-State.

9.5 BOUNDARY SCAN REGISTER

The Boundary Scan Register path from TDI to TDO is as follows:

1	PHDAT(7)	IN	CMOS
2	PHDAT(6)	IN	CMOS
3	PHDAT(5)	IN	CMOS
4	PHDAT(4)	IN	CMOS
5	PHDAT(3)	IN	CMOS
6	PHDAT(2)	IN	CMOS
7	PHDAT(1)	IN	CMOS
8	PHDAT(0)	IN	CMOS
9	TABORT	IN	CMOS
10	ADDR16	IN	CMOS
11	LDADDR	IN	CMOS
12	TXRDY	IN	CMOS
13	TPATH(7)	OUT	CMOS
14	TPATH(6)	OUT	CMOS
15	TPATH(5)	OUT	CMOS
16	TPATH(4)	OUT	CMOS
17	TPATH(3)	OUT	CMOS
18	TPATH(2)	OUT	CMOS
19	TPATH(1)	OUT	CMOS
20	TPATH(0)	OUT	CMOS
21	TPRITY	OUT	CMOS
22	TXCTL(1)	OUT	CMOS
23	TXCTL(0)	OUT	CMOS
24	RABORT	OUT	CMOS
25	MATCHO	OUT	CMOS
26	RCCTL(4)	IN	CMOS
27	RCCTL(3)	IN	CMOS
28	RCCTL(2)	IN	CMOS
29	RCCTL(1)	IN	CMOS
30	RCCTL(0)	IN	CMOS
31	REJECT	IN	CMOS
32	RPATH(7)	IN	CMOS
33	RPATH(6)	IN	CMOS
34	RPATH(5)	IN	CMOS
35	RPATH(4)	IN	CMOS
36	RPATH(3)	IN	CMOS
37	RPATH(2)	IN	CMOS
38	RPATH(1)	IN	CMOS
39	RPATH(0)	IN	CMOS
40	RPRITY	IN	CMOS
41	RESET	IN	TTL
42	BYTCLK	IN	TTL
43	SYMCLK	IN	TTL
44	BCNTL(3)	IN	TTL
45	BCNTL(2)	IN	TTL
46	BCNTL(1)	IN	TTL
47	BCNTL(0)	IN	TTL
48	BRW	IN	TTL
49	ACNTL(3)	IN	TTL
50	ACNTL(2)	IN	TTL
51	ACNTL(1)	IN	TTL
52	ACNTL(0)	IN	TTL
53	ARW	IN	TTL
54	ADATA(31)	I/O	TTL
55	ADATA(30)	I/O	TTL
56	ADATA(29)	I/O	TTL
57	ADATA(28)	I/O	TTL
58	ADATA(27)	I/O	TTL
59	ADATA(26)	I/O	TTL
60	ADATA(25)	I/O	TTL
61	ADATA(24)	I/O	TTL
62	ADATA(23)	I/O	TTL
63	ADATA(22)	I/O	TTL
64	ADATA(21)	I/O	TTL
65	ADATA(20)	I/O	TTL
66	ADATA(19)	I/O	TTL
67	ADATA(18)	I/O	TTL
68	ADATA(17)	I/O	TTL
69	ADATA(16)	I/O	TTL
70	ADATA(15)	I/O	TTL
71	ADATA(14)	I/O	TTL
72	ADATA(13)	I/O	TTL
73	ADATA(12)	I/O	TTL
74	ADATA(11)	I/O	TTL
75	ADATA(10)	I/O	TTL
76	ADATA(9)	I/O	TTL
77	ADATA(8)	I/O	TTL
78	ADATA(7)	I/O	TTL
79	ADATA(6)	I/O	TTL
80	ADATA(5)	I/O	TTL
81	ADATA(4)	I/O	TTL
82	ADATA(3)	I/O	TTL
83	ADATA(2)	I/O	TTL
84	ADATA(1)	I/O	TTL
85	ADATA(0)	I/O	TTL
86	APRTY(3)	I/O	TTL
87	APRTY(2)	I/O	TTL
88	APRTY(1)	I/O	TTL
89	APRTY(0)	I/O	TTL
90	AREQ(1)	OUT	TTL
91	AREQ(2)	OUT	TTL
92	A-IEN	DRC	Int
93	A-OEN	TSC	Int
94	AREQ(0)	OUT	TTL
95	AREQ(3)	OUT	TTL
96	ACS(1)	IN	TTL
97	ACS(0)	IN	TTL
98	BCS(0)	IN	TTL
99	BCS(1)	IN	TTL
100	BREQ(3)	IN	TTL
101	BREQ(0)	IN	TTL
102	B-OEN	TSC	Int
103	B-IEN	DRC	Int
104	BREQ(2)	OUT	TTL
105	BREQ(1)	OUT	TTL
106	BPRTY(0)	I/O	TTL
107	BPRTY(1)	I/O	TTL
108	BPRTY(2)	I/O	TTL
109	BPRTY(3)	I/O	TTL
110	BDATA(0)	I/O	TTL
111	BDATA(1)	I/O	TTL
112	BDATA(2)	I/O	TTL
113	BDATA(3)	I/O	TTL
114	BDATA(4)	I/O	TTL
115	BDATA(5)	I/O	TTL
116	BDATA(6)	I/O	TTL
117	BDATA(7)	I/O	TTL
118	BDATA(8)	I/O	TTL
119	BDATA(9)	I/O	TTL
120	BDATA(10)	I/O	TTL
121	BDATA(11)	I/O	TTL
122	BDATA(12)	I/O	TTL
123	BDATA(13)	I/O	TTL
124	BDATA(14)	I/O	TTL
125	BDATA(15)	I/O	TTL
126	BDATA(16)	I/O	TTL
127	BDATA(17)	I/O	TTL
128	BDATA(18)	I/O	TTL
129	BDATA(19)	I/O	TTL
130	BDATA(20)	I/O	TTL
131	BDATA(21)	I/O	TTL
132	BDATA(22)	I/O	TTL
133	BDATA(23)	I/O	TTL
134	BDATA(24)	I/O	TTL
135	BDATA(25)	I/O	TTL
136	BDATA(26)	I/O	TTL
137	BDATA(27)	I/O	TTL
138	BDATA(28)	I/O	TTL
139	BDATA(29)	I/O	TTL
140	BDATA(30)	I/O	TTL
141	BDATA(31)	I/O	TTL
142	AINT	OUT	TTL
143	BINT	OUT	TTL

SECTION 10 ELECTRICAL SPECIFICATIONS

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the clock(s) and possibly to one or more other signals.

10.1 MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	–0.3 to +7.0	V
Input Voltage	V_{in}	–0.3 to +7.0	V
Operating Temperature Range MC68339	T_A	 0 to 70	°C
Storage Temperature Range	T_{stg}	–55 to +150	°C

This device contains protective circuitry against damage due to high static voltages or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either GND or V_{CC}).

10.2 THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance for PGA	θ_{JA}	20	°C/W
	θ_{JC}	TBD	°C/W
Thermal Resistance for CQFP	θ_{JA}	TBD	°C/W
	θ_{JC}	TBD	°C/W

10.3 FSI DC ELECTRICAL SPECIFICATIONS

(V_{CC} = 5.0 V_{dc} ± 10%, GND = 0 V_{dc}, T_A = 0°C to 70°C)

Symbol	Characteristics	Min	Typical	Max	Unit
V_{IH}	Input High Voltage (TTL pins)	2.0		Vcc	V
V_{IL}	Input Low Voltage (TTL pins)	GND – 0.3		0.8	V
V_{IHC}	Input High Voltage (CMOS pins)	0.7 • V_{CC}		V_{CC}	V
V_{ILC}	Input Low Voltage (CMOS pins)	GND – 0.3		0.2 • V_{CC}	V
I_{in}	Input Leakage Current	—		20	µA
C_{in}	Input Capacitance All Pins	—		15	pF
I_{tsi}	Three State Leakage Current	—		20	µA
V_{OH}	Output High Voltage (I_{OH} = 400µA)	V_{CC} – 1.0		—	V
V_{OL}	Output Low Voltage (I_{OL}= 5.3 mA) ADATAx, APRTYx, BDATAx, BPRTYx (I_{OL} = 3.2 mA) AREQx, BREQx, TDO, AINT, BINT (I_{OL} = 1.9 mA) TPATHx, TPRITY, TXCTLx, RABORT, MATCHO	— — —		0.5 0.5 0.3 • V_{CC}	V
P_D	Power Dissipation	1.1@Vcc	1.5@5.5V	1.65@5.5V	Watts
I_{DD}	Operating Current	220@Vcc	270@5.5V	300@5.5V	mA

10.4 POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \qquad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts—Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins—User Determined

For most applications, $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \qquad (2)$$

Solving Equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \qquad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at thermal equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

The total thermal resistance of a package (θ_{JA}) can be separated into two components, θ_{JC} and θ_{CA}, representing the barrier to heat flow from the semiconductor junction to the package (case) surface (θ_{JC}) and from the case to the outside ambient air (θ_{CA}). These terms are related by the equation:

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \qquad (4)$$

θ_{JC} is device related and cannot be influenced by the user. However, θ_{CA} is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling, and thermal convection. Thus, good thermal management on the part of the user can significantly reduce θ_{CA} so that θ_{JA} approximately equals θ_{JC}. Substitution of θ_{JC} for θ_{JA} in equation (1) results in a lower semiconductor junction temperature.

Values for thermal resistance presented in this document, unless estimated, were derived using the procedure described in Motorola Reliability Report 7843, "Thermal Resistance Measurement Method for MC68XX Micro component Devices," and are provided for design purposes only. Thermal measurements are complex and dependent on procedure and setup. User derived values for thermal resistance may differ.

10.5 CLOCKS AND MAC INTERFACE TIMING

Num	Characteristics	Min	Max	Unit
T1	BYTCLK Cycle Time	80	—	ns
T2	SYMCLK Cycle Time	40	—	ns
T3	SYMCLK Pulse Width Low	16	-	ns
T4	SYMCLK Pulse Width High	16	-	ns
T5	SYMCLK to BYTCLK High Skew	0	15	ns
T6	SYMCLK to BYTCLK Low Skew	0	15	ns
T7	SYMCLK Rise or Fall Time	—	4	ns
T8	Signal Valid to SYMCLK Falling Edge[1]	0	—	ns
T8A	Signal Valid to SYMCLK Falling Edge[1]	13	—	ns
T9	SYMCLK Falling Edge to Signal Invalid[1]	3	—	ns
T9A	SYMCLK Falling Edge to Signal Invalid[1]	4	—	ns
T10	BYTCLK Rising Edge to Signal Valid[1,2]	—	25	ns

NOTES:
1. Relative to the falling edge of SYMCLK immediately preceding the rising edge of BYTCLK.
2. For every 10-pF loading capacitance more than 50-pF, add 1 ns.
3. Note that rise and fall times are not measured
4. This timing is for low frequencies, for 25MHz, it is limited to 3ns.

NOTE

The FSI samples its input signals from the MAC with the falling edge of SYMCLK, which precedes the rising edge of BYTCLK. This is consistent with the MAC which outputs its signals to the FSI on the rising edge of BYTCLK. The FSI outputs and MAC inputs operate in a similar fashion.

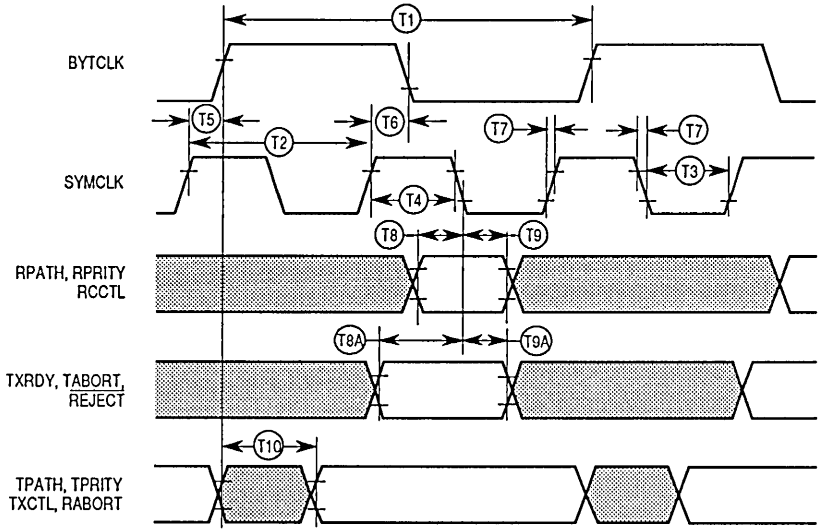


Figure 10-1. FSI Clocks and MAC Interface Timing

10.6 SYSTEM INTERFACE READ AND WRITE TIMING

Num	Characteristics	Min	Max	Unit
T21	Chip Select Width Asserted	18	—	ns
T22	Chip Select Width Negated	18	—	ns
T22A	Chip Select Width Negated for Other Port Address Access (CNTLx = 1000)	10	—	ns
T23	Control Valid to Chip Select Asserted	12	—	ns
T24	Chip Select Asserted to Control Invalid	0	—	ns
T25	R/W Valid to Chip Select Asserted	10	—	ns
T26	Chip Select Asserted to R/W Invalid	0	—	ns
T27	Data_In Valid to Chip Select Negated	5	—	ns
T27A	Data_In Valid to Chip Select Negated with Parity Checking	10	—	ns
T28	Chip Select Negated to Data_In Invalid	3	—	ns
T30	Chip Select Asserted to Data_Out Valid for Data Accesses	—	15	ns
T30A	Chip Select Asserted to Data_Out Valid for Address Accesses	—	15	ns
T30B	Chip Select Asserted to Data_Out Valid for Register Accesses	—	15	ns
T30C	Chip Select Asserted to Parity_Out Valid for Data Accesses	—	15	ns
T30D	Chip Select Asserted to Parity_Out Valid for Address Accesses	—	17	ns
T30E	Chip Select Asserted to Parity_Out Valid for Register Accesses	—	17	ns
T31	Chip Select Negated to Data_Out Three State	1	13	ns
T32	Chip Select Asserted to Request Valid	—	12	ns
T33	ACSx(BCSx) Asserted for Data Access (ACNTLx(BCNTLx) = 0010) to BCSx(ACSx) Asserted for Other Port Address Accesses (BCNTLx(ACNTLx) = 1000)	15	—	ns

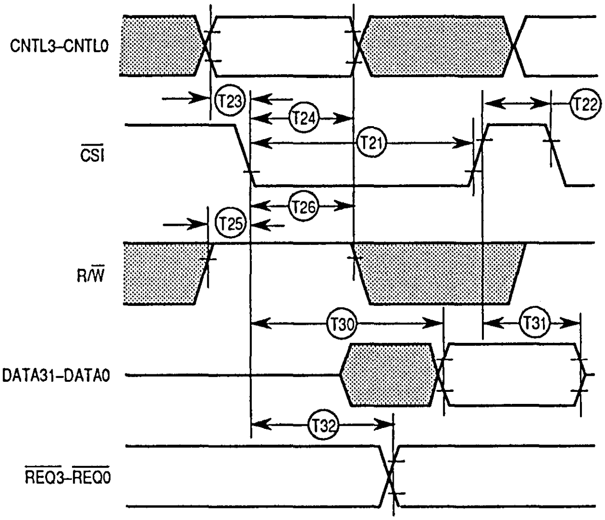


Figure 10-2. FSI Read Timing

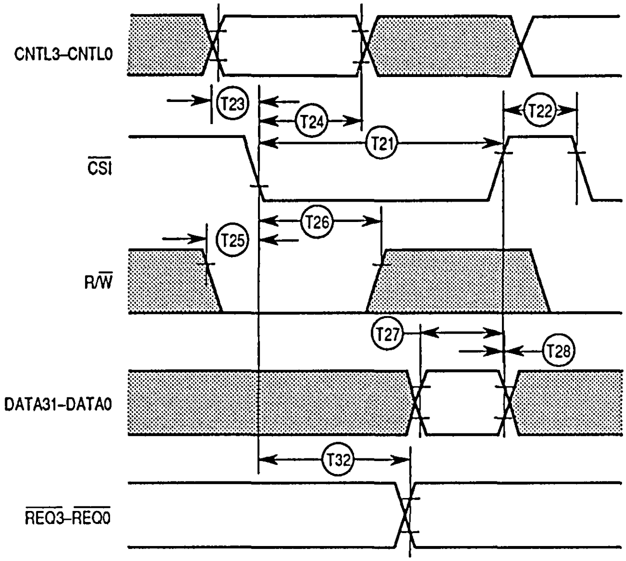


Figure 10-3. FSI Write Timing

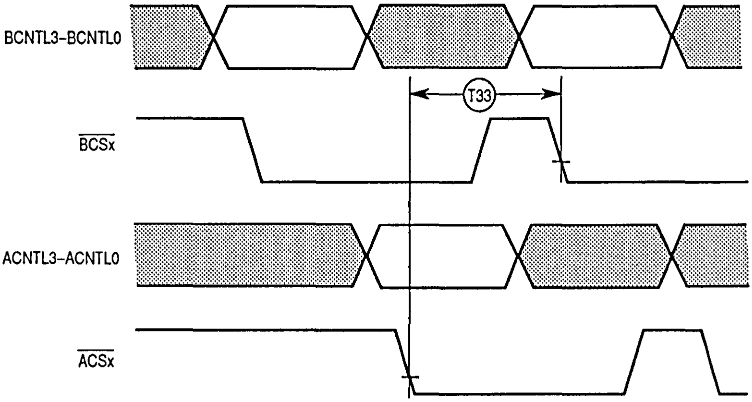


Figure 10-4. FSI Nonmultiplexed Two-Port Timing

10.7 CAM INTERFACE TIMING

Num	Characteristics	Min	Max	Unit
T41	LDADDR Asserted to SYMCLK Falling Edge[1]	0	—	ns
T42	SYMCLK Falling Edge to LDADDR Invalid[1]	3	—	ns
T43	ADDR16 Asserted to SYMCLK Falling Edge[1]	0	—	ns
T44	SYMCLK Falling Edge to ADDR16 Invalid[1]	3	—	ns
T45	BYTCLK Rising Edge to MATCHO Asserted[2]	—	25	ns
T46	PHDAT Valid to SYMCLK Falling Edge[1]	0	—	ns
T47	SYMCLK Falling Edge to PHDAT Invalid[1]	3	—	ns

NOTES:
1. Relative to the falling edge of SYMCLK when BYTCLK is low.
2. For every 10-pF loading capacitance more than 50-pF, add 1 ns.

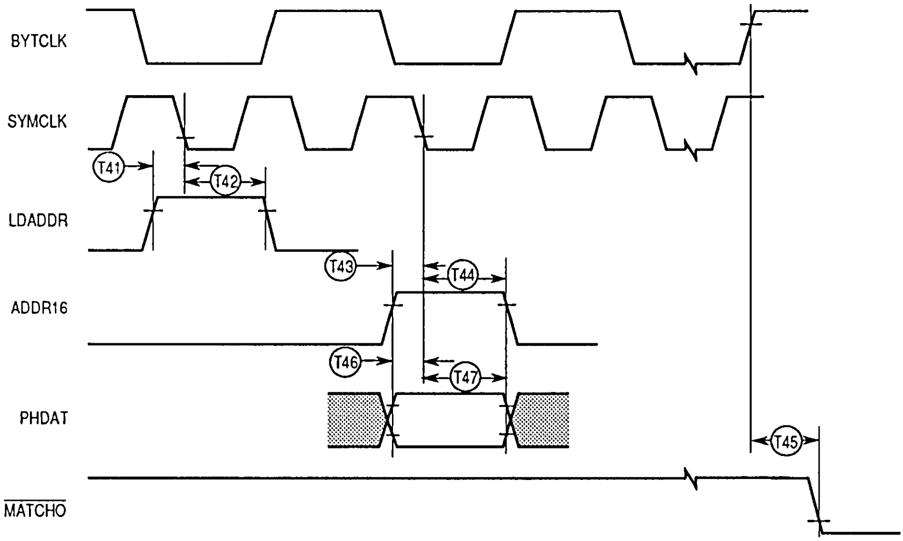


Figure 10-5. FSI-CAM Interface Timing

10.8 REJECT TIMING

When REJECT is asserted, the FSI asserts RABORT to the MAC on the next rising edge of BYTCLK. As a result, the MAC will supply the end data indication to the FSI.

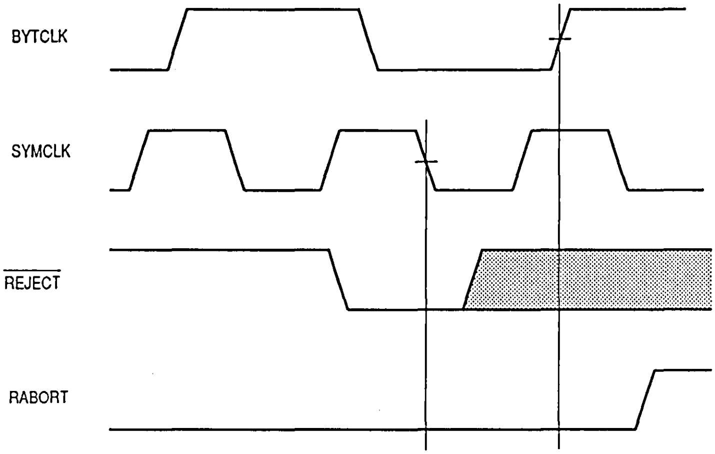


Figure 10-6. FSI REJECT/RABORT Timing

10.9 JTAG TIMING

Num	Characteristics	Min	Max	Unit
T61	TDI Valid to TCK Rising Edge	2	—	ns
T62	TMS Valid to TCK Rising Edge	6	—	ns
T63	TCK Asserted to TDI Invalid	1	—	ns
T64	TCK Asserted to TMS Invalid	0	—	ns
T65	TCK Negated to TDO Valid[1]	—	15	ns

NOTE: 1. For every 10 pF Loading capacitance more than 50-pF, add 1 ns.

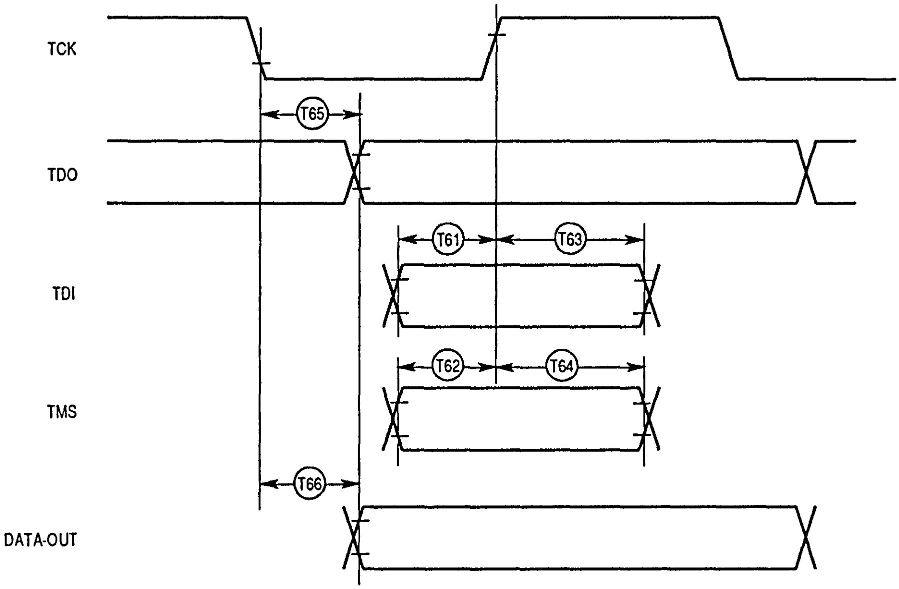


Figure 10-7. JTAG Timing

SECTION 11
ORDERING INFORMATION AND MECHANICAL DATA

This section contains ordering information, pin assignments, and package dimensions for the MC68839 FSI.

11.1 ORDERING INFORMATION

Package Type	Frequency	Temperature	Order Number
Pin Grid Array (PGA)	25 MHz	0–70°C	XC68839RC
Ceramic Surface Mount (CQFP)	25 MHz	0–70°C	XC68839FE

11.2 PIN ASSIGNMENTS

11.2.1 Pin Grid Array (PGA)

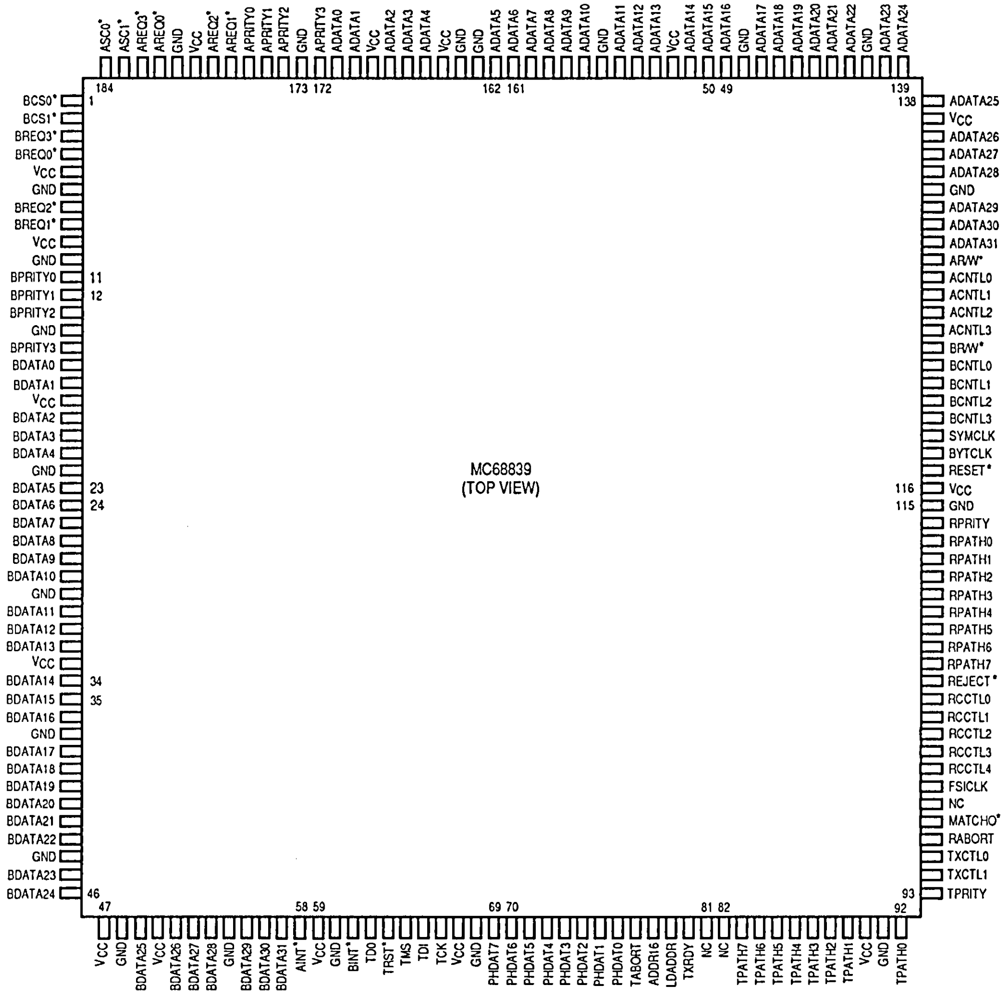
MC68839 (BOTTOM VIEW)

	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
A	BDATA28	BDATA25	BDATA22	BDATA19	BDATA17	BDATA14	BDATA12	BDATA11	BDATA9	BDATA7	BDATA5	BDATA3	BDATA2	BDATA0	BPRITY1	BPRITY0	BCS1*
B	BDATA31	BDATA27	BDATA23	BDATA21	BDATA18	BDATA15	BDATA13	BDATA10	BDATA8	BDATA6	BDATA4	BDATA1	BPRITY3	BREQ0*	BREQ3*	ACS0*	ACS1*
C	BINT*	BDATA 30	BDATA26	BDATA24	BDATA20	BDATA16	VCC	VCC	VCC	GND	VCC	BPRITY2	BREQ1*	BREQ2*	BCS0*	AREQ3*	APRITY0
D	TRST*	AINT*	BDATA29				GND	GND		GND	GND				AREQ0*	AREQ2*	APRITY1
E	TCK	TMS	TDO												AREQ1*	APRITY2	ADATA0
F	PHDAT6	PHDAT7	TDI												APRITY3	ADATA1	ADATA2
G	PHDAT4	PHDAT5	VCC	GND										GND	VCC	ADATA3	ADATA4
H	PHDAT3	VCC	GND	GND										GND	VCC	ADATA5	ADATA6
J	PHDAT2	VCC	VCC												VCC	ADATA8	ADATA7
K	PHDAT1	PHDAT0	GND	GND										GND	GND	ADATA10	ADATA9
L	TABORT	ADDR16	VCC	GND										GND	VCC	ADATA12	ADATA11
M	LDADDR	TXRDY	NC												ADATA16	ADATA14	ADATA13
N	NC	TPATH7	TPATH5												ADATA20	ADATA18	ADATA15
P	TPATH6	TPATH4	TPATH2				GND	GND		GND	GND				ADATA23	ADATA21	ADATA17
R	TPATH3	TPATH1	TPRITY	MATCHO*	RCCTL3	REJECT*	RPATH5	GND	VCC	VCC	BCNTL2	BR/W*	ACNTL0	ADATA29	ADATA25	ADATA24	ADATA19
T	TPATH0	TXCTL1	RABORT	FSICLK	RCCTL1	RPATH7	RPATH4	RPATH2	RPRITY	BYTCLK	SYMCLK	BCNTL0	ACNTL2	AR/W*	ADATA28	ADATA26	ADATA22
U	TXCTL0	NC	RCCTL4	RCCTL2	RCCTL0	RPATH6	RPATH3	RPATH1	RPATH0	RESET*	BCNTL3	BCNTL1	ACNTL3	ACNTL1	ADATA31	ADATA30	ADATA27

NOTE

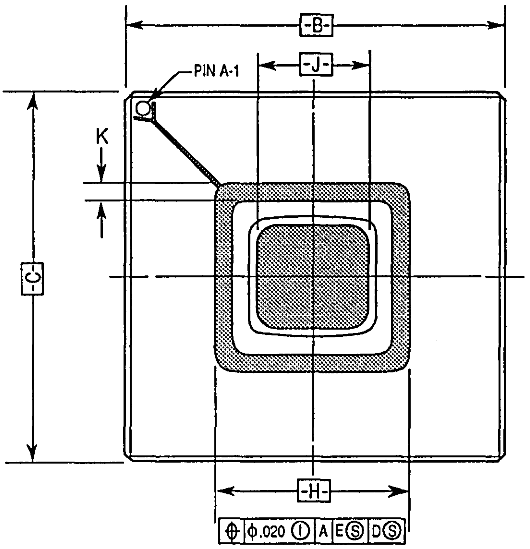
Pin J14 will be added as a KEY pin to future FSI chips. This pin may be used as a GND pin or broken off.

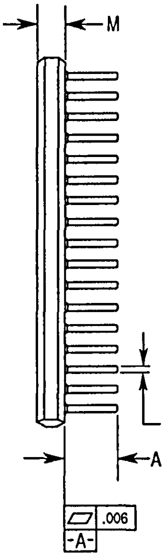
11.2.2 CERAMIC SURFACE MOUNT (CQFP)

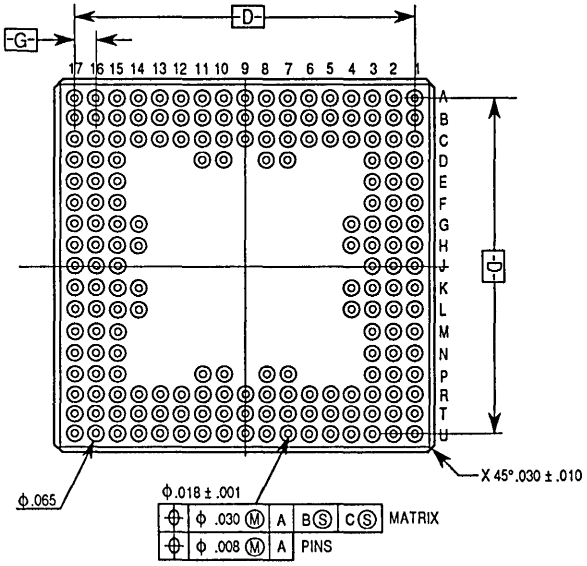


11.3 PACKAGE DIMENSIONS

184 PGA
CASE To Be Determined
(Preliminary)

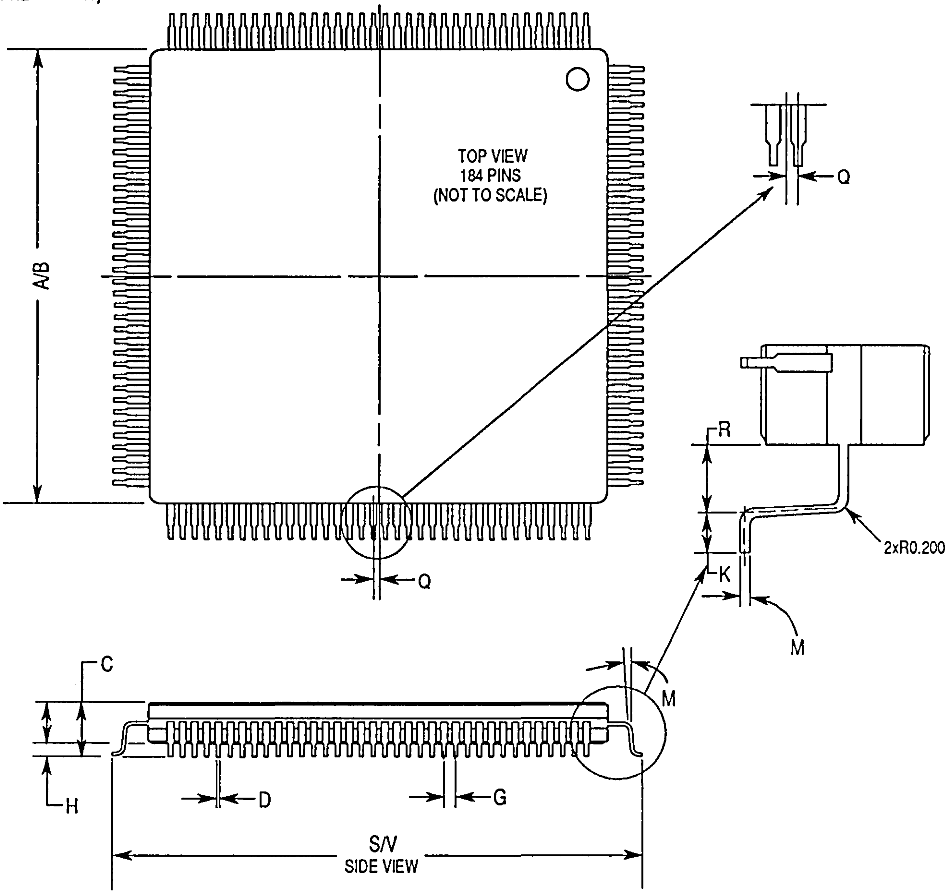






DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A			0.17	0.19
B			1.746	1.774
C			1.746	1.774
D			1.6	1.6
G			.10	.10
H			.95	.97
J			.578	.59
K			0.070	0.070
M			0.9	0.11

184 PIN QFP
CASE TO BE DETERMINED
(PRELIMINARY)



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	30.98	32.85		
B	30.98	32.85		
C	3.27	4.58		
D	0.22	0.41		
G	0.65	—		—
H	0.25	0.88		
J	0.13	0.25		
K	0.65	0.95		
M	0°	8°	0°	8°
Q	0.325	—		—
R	0.50	—		—
S	34.95	35.45		
V	34.95	35.45		

APPENDIX A
BUS CONTROL LOGIC EXAMPLE

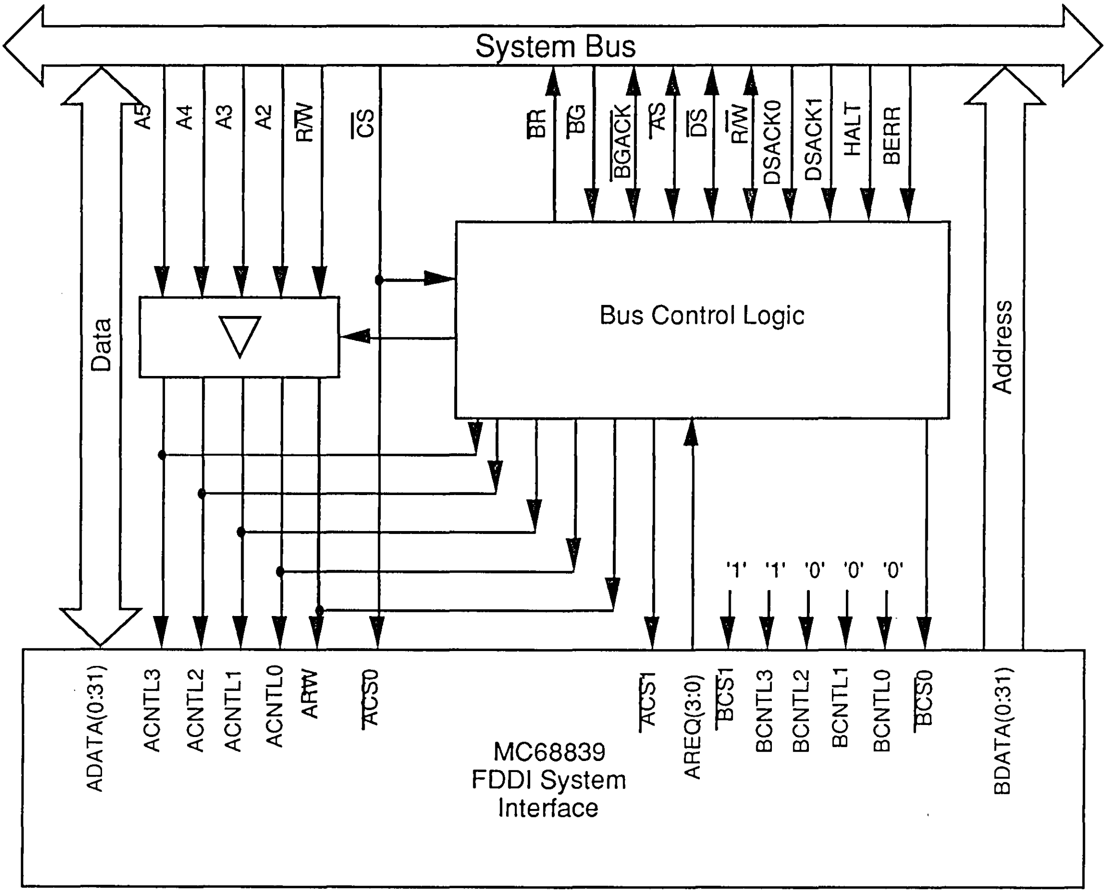


Figure A-1. MC68020 Bus Control Logic Signals

APPENDIX B SYSTEM CONFIGURATIONS

The FDDI System Interface allows for a number of different FDDI system configurations. These different configurations allow the user to select the most appropriate configuration for the application. The different configurations are shown below and are chosen by the user based on system latency, functional task division, bus width, and bus format (multiplexed/non-multiplexed).

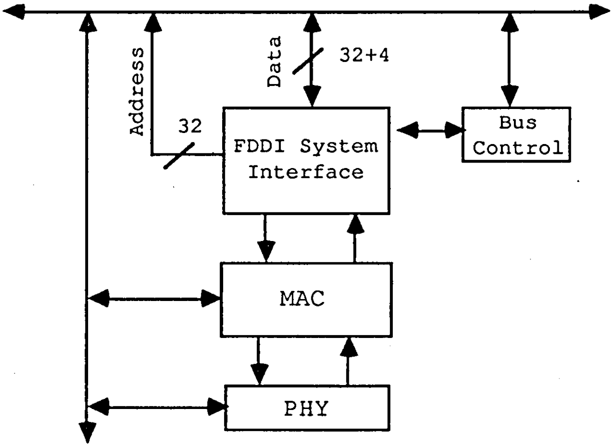


Figure B-1. Basic Configuration

This configuration can be used when the FDDI node is either on a separate board from the main system or is on the same bus as the main processor. If it is a separate board, the bus shown may be a back plane bus and the latency of the system bus should be short enough to allow the FSI to operate with no local memory. If the bus shown is the main or a local microprocessor bus, this configuration assumes the bus can give the FDDI system enough of its bandwidth to handle FDDI traffic. In this case, the bus shown can be the system bus.

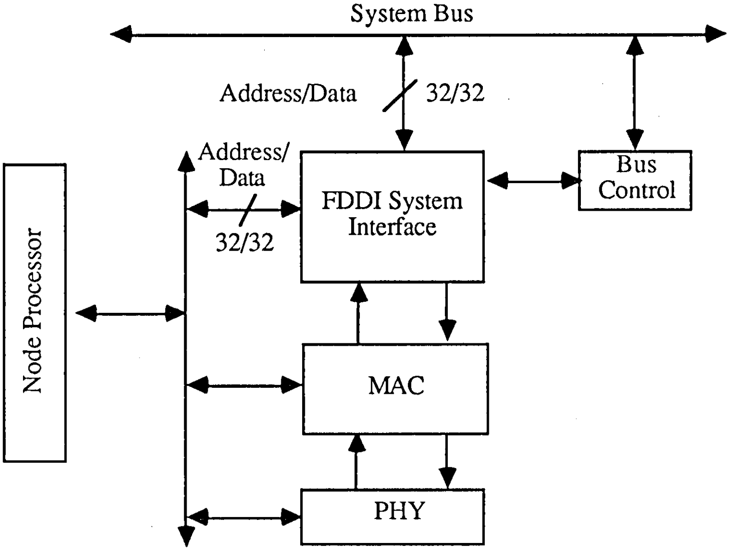


Figure B-2. Basic Configuration With Separate Node Processor

The configuration in Figure B2 can be used when the user wants to off load FDDI specific tasks (such as SMT) and the control of the MAC and PHY chips to a special node processor. These FDDI-specific tasks may include the handling of station management (SMT) and synchronous frames, and monitoring statistics and the network topology. Because these tasks are neither very frequent nor very complex, and because the FDDI System Interface contains the ability to store these frames separately, the node processor may be simple and inexpensive. In this configuration, the data frames are loaded directly from the system memory .

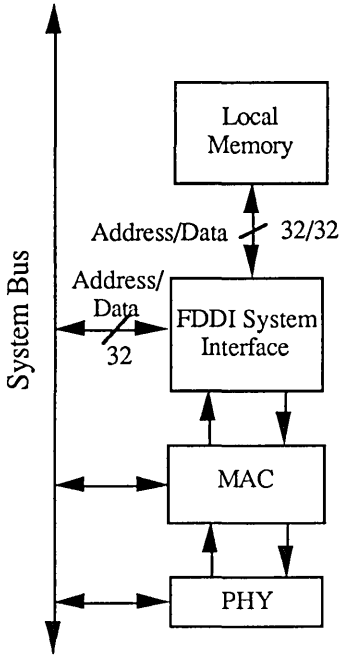


Figure B3. Basic Configuration with Local Memory

In this configuration, the latency of the processor bus may be insufficient to handle the FDDI network traffic without more buffering than that which is provided in the FDDI System Interface. A local memory may be added to handle this additional buffering requirement. The second port may be used for the host to directly access the FDDI System Interface to give it commands, read its status, and to handle the transmission of Immediate frames. With this configuration, the host processor must handle the control of the MAC and PHY, and the operation of SMT.

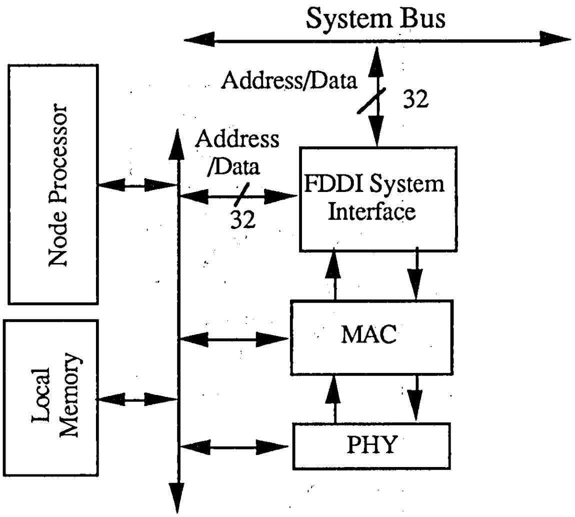


Figure B-4. Local Memory with Separate Node Processor

In this configuration, local memory has been added because the latency of the host processor bus cannot handle the FDDI network traffic without more buffering than that which is provided in the FDDI System Interface. The node processor may be used to off load FDDI-specific tasks and the control of the MAC and PHY chips. These FDDI-specific tasks may include the handling of station management (SMT) and synchronous frames, and the monitoring of statistics and of network topology.

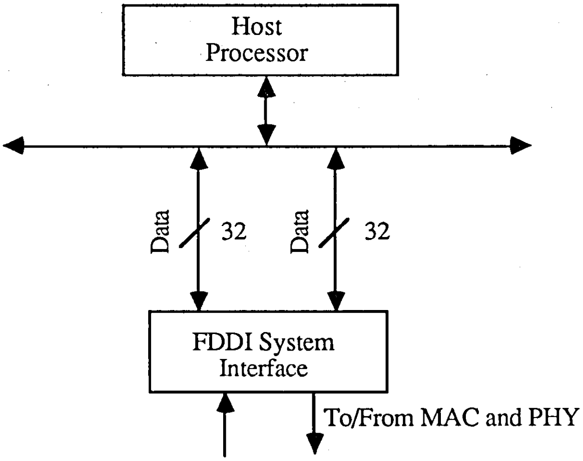


Figure B-5. Slave Configuration With 64-Bit Data

In this configuration, the FDDI System Interface is a slave to the host processor. This means that either the system has its own DMA or the host handles the transfer of data over its bus and the FDDI System Interface's DMA is not used. The FDDI System Interface connects directly to the host processor bus which must be fast enough to fill and empty the FIFOs of the FDDI System Interface. The 64-bits of data enable the FDDI System Interface to be connected to specialized system back planes and new generation, high-performance processors with 64-bit data busses.

This configuration may be used in very high performance applications and may be sufficient for handling full duplex communication protocols. Even if the network frequency increases, the FDDI System Interface would still be able to handle the traffic.

Note that 64-bit data configurations are also possible with the basic configuration of Figure B1.

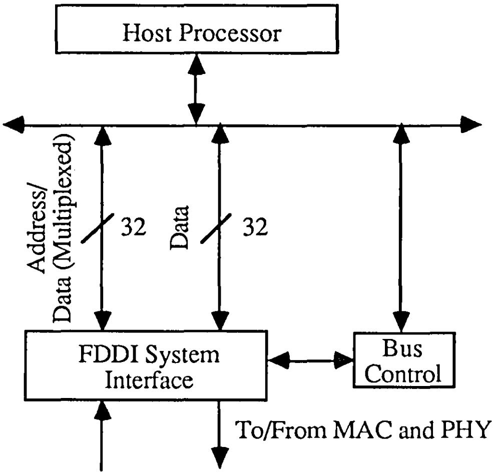


Figure B-6. 64-Bit Data With FSI DMA

In the DMA configuration with 64-bit data, the FDDI System Interface is connected to a very high performance microprocessor. In this case, there is no system-wide DMA capability, so the DMA of the FDDI System Interface is used. Port A Data Register holds the most significant portion of the data and the Port B Data Register holds the least significant portion of the data. The address is generated in Port A but can be accessed via either port.

APPENDIX C
PERFORMANCE REQUIREMENTS

This appendix is intended to give a better understanding of what the performance requirements are for the different configurations available with the FSI.

The network speed is 100 Mbit/sec or 80ns/byte. In order to be able to transmit and receive frames consecutively, the system bus must have a cycle time as calculated in this section. The variable *K*, used in the following examples is the coefficient of average bus speed in byte transfer units of 80ns. *K* may be viewed as:

K = network speed (12.5 MBytes per second)/bus speed (Bytes per second)

For example, in case of bus width = 32bit and bus cycle = 160ns, *K* will be equal to 0.5, i.e., the bus speed is twice the network speed.

The internal overhead of the FSI, *O*, before it will start the data transfer is ~1280 ns. This time in byte transfer units (80ns) is 16 bytes. This overhead is applied whenever the FSI changes between ring descriptor accesses and data DMA accesses. This makes the total time needed for DMA of *X* bytes:

Total data DMA time = {O + KX} or 16 + KX Eq. 1

In our examples we attempt to show the transmission and reception of back to back frames with a minimum inter-frame spacing of 8 bytes of Idle symbols. If one assumes that the CRC, or FCS field, of a data frame does not have to be received, then the network overhead for both transmitted and received frames is 12 bytes. That is, the network, or in our case the MAC, imposes additional symbols which are not a part of the DMA'd data frame such as the Idle symbols, JK, ED, etc. Therefore, the Network time for every data frame is the number of frame bytes including FC, DA, SA and info field plus the overhead, or:

{Network time} = X + 12 Eq. 2

The information field of each frame with 48-bit addresses is:

I = X - (PRH+ FC + DA + SA) or I = X - 16.

During the transmission or reception of back to back data frames the general equation for bus utilization is given as:

U = {System time} / {Network time} Eq. 3

In the following subsections the user is provided with formulae which describe the transmission and reception of back to back frames. The user may determine the bus utilization for any given frame size or amount of data to be transferred. Alternatively, the user may determine the amount of data or frame size to implement to achieve any given rate of bus utilization.

C.1 FSI TRANSMIT PERFORMANCE

The FSI has room inside its internal memory for as many as 20 descriptors. Also, the Tx DMA operation has priority higher than Descriptor input operation. Therefore, in case of small frames (frame length is smaller than a watermark) and assuming each frame to have one descriptor, the internal activities of the FSI will be as following:

a input 20 descriptors (of 20 frames)

b DMA 20 frames

c go to (a)

Note, that indication write activities are not taken into account in this analysis.

There are 8 bytes in one descriptor. Only one period of internal overhead (16 byte times) is used for 20 descriptors. The total time is in byte transfer units (80ns). The total time needed for reading 20 Descriptors is :

Total Descriptor read time = {16 + 20K8} Eq. 4

C.1.1 Case 1. Using same port for data and descriptors

For applications where both data and descriptors are read from the same port (note that this is the only configuration when using 64 bit data width) the equation will be:

{System time} = {Total data DMA time} + {Total Descriptor read time}

or $F(X + N) = F(O + KX) + (O + FKD)$

where:

F is the number of frames, here 20

O is the FSI overhead, 16 bytes

N is the Network overhead of 12 bytes

D is the number of bytes in a descriptor, 8

K is the bus coefficient and X is the amount of data in a frame

Computing for 20 data frames, and thereby 20 descriptors, with *X* data in each frame and a total network time as given above yields:

$$20\,(16 + KX) + (16 + 20\,(K\,(8))) = 20\,(X + 12)$$

Solving for *X*:

$$X = (24 + K40) / (5\,(1 - K))$$

The external bus utilization should take into account all the indications which also need to be written back to the system memory, therefore, following equation 3:

U = {data DMA time} + {Indication write time} + {Descriptor read time} / {Network time}

$U = (KX + K8 + K8) / (X+12)$

$U = K (X + 8 + 8) / (X + 12) = K (X + 16) / (X + 12)$

For example:

A. assuming the average access time to be 160ns for 32 bit word, i.e., $K = 0.5$:

$X = (24+20) / 2.5= 44 / 2.5 = 18$

$U = 0.5 (18+16) / (18+12) = 0.567$ or 56.7%

Therefore, in this case the minimum frame length should be 18 bytes or 2 bytes of INFO in order to transmit 20 data frames back to back.

B. assuming the average access time to be 160ns for 64 bit word, i.e., $k = 0.25$

$X = (24+10) / 3.75 = 10$

$U = 0.25 (10+16) / (10+12) = 0.295$ or 29.5%

Therefore, in this case the minimum frame length should be 10 bytes total to transmit 20 data frames back to back.

C.1.2 Case 2. Using different ports for data and descriptors

For application when different ports are used for data and descriptor the equation will take into account only the DMA overhead, therefore it follows:

$\{16 + KX\} = X + 12$ following Equation 4

or

$X = 4 / (1 - K)$

The external bus utilization for data only is calculated as follows:

$U = KX / (X+12)$

For example assuming the average access time to be 160ns for 32 bit word, i.e., $k = 0.5$

$X = 4 / 0.5= 8$

$U = 0.5 (8 / (8+12)) = 0.20$ or 20%

Therefore, in this case the minimum frame length should be 16 bytes or ZERO bytes of INFO.

C.2 FSI RX OVERHEAD

In the receive case the receive buffer descriptors are read when the ring has been defined and made ready.

Indications for each frame received are written back out to memory as the conditions warrant and as the internal priority scheme of the FSI's internal tasks allow. Thus, some receive indications may be written out individually, others may be written out in groups. The total time needed for a single indication (there are 8 bytes in one indication) to be written is :

{20 + K8}

There are two different cases which matter for performance analysis of the FSI as described below.

C.2.1 Case 1. Using same port for data and indications

For application when both data and indication are output from the same port (note that this is the only configuration when using 64 bit data width) the system time equation will be:

{System time} = {Total DMA time} + {Total Ind. time}

{Total DMA time} + {Total Ind. time} = {Network time}

{20 + KX} + {20 + K8} = X + 12

Using the equation above we obtain

X = (28 + K8) / (1-K)

The external bus utilization should take into account another 8 bytes of descriptor. Therefore, the general equation for external bus utilization will be:

U = {DMA time} + {Ind. time} + {Desc. time} / {Network time} following equation 3

U = K (X + 8 + 8) / (X + 12) = K (X + 16) / (X + 12)

For example:

A. assuming the average access time to be 160ns for 32 bit word, i.e., k = 0.5

X = (28 + 4) / 0.5= 64

U = 0.5 (64 + 16) / (64 + 12) = 0.53 or 53%

Therefore, in this case the minimum frame length should be 64 bytes or 48 bytes of INFO for continuous reception.

B. assuming the average access time to be 160ns for 64 bit word, i.e., k = 0.25

X = (28 + 2) / 0.75= 40

U = 0.25 (40 + 16) / (40 + 12) = 0.27 or 27%

Therefore, in this case the minimum frame length should be 40 bytes or 24 bytes of INFO for continuous reception.

C.2.2 Case 2. Using different ports for data and indications

For application when different ports are used for data and indication the equation will take into account only the DMA overhead,.therefore it will be as following:

$\{20 + KX\} = X + 12$; or

$X = 8 / (1 - K)$

The external bus utilization for data only will be calculated as follows:

$U = KX / (X + 12)$

For example:

Assuming the average access time to be 160ns for 32 bit word, i.e., $K = 0.5$

$X = 8 / 0.5 = 16$

$U = 0.5 (16) / (16 + 12) = 0.29$ or 29%

Therefore, in this case the minimum frame length should be 16 bytes or ZERO bytes of INFO for continuous reception.

C.3 LOW PERFORMANCE SYSTEMS

In low performance systems, the requirements from transmitter and receiver are much less. However, this system should be able to transmit the entire frame on the FDDI network even if the bus bandwidth allocated for the FSI is insufficient. In this case, the use of Local Memory will alleviate the performance stress on the FDDI system, allowing slower system busses to easily connect to the IFDDI. Local Memory offers greater "temporary" storage for FDDI frames as the frames come in from the MAC interface, essentially increasing the internal receive data FIFO storage space.

C.4 NODE PROCESSOR

It may also be useful to implement a Node Processor to take care of all the Station Management traffic. This processor does not need to have as much performance as the host processor because the throughput of management frames is low and their frames are short; i.e., the FIFO used to transmit them may be set with a watermark greater than the maximum frame size. It is not required to have a node processor. The chip set handles enough of the real time portions of SMT such that very little of the host processor is needed to run SMT. The decision of whether or not to have a separate node processor is an architectural preference.

APPENDIX D
RING MEMORY STRUCTURE AND THE FSI IMPLEMENTATION

This discussion concerns use of ring memory structure in a high speed environment.

RING MEMORY STRUCTURE: GENERAL DISCUSSION

After a ring is set up (any number of descriptors) up to X valid descriptors will be read into the ring memory structure regardless of how large the ring is. "X" is a number dependant on the amount of internal memory devoted to temporary storage of descriptors, and depends on the particular implementation.

If the ring is less than X descriptors, there must be one invalid entry in that ring. Otherwise the ring may wrap while reading the descriptors into internal memory, filling up the rest of the internal memory that is devoted to temporary storage of descriptors. This is particular to ring memory structure using any small ring.

The transmit is then performed from the internal copy, greatly reducing transmission latency.

A small ring in the MC68839 FSI implementation is defined as any ring that does not fill the 20 descriptor-bounded internal memory. If a ring that is smaller than 20 descriptors is used, there must be one invalid entry (own bit reset/false) in that ring. Otherwise the ring will wrap while reading the descriptors into internal memory and multiple copies of the ring may be transmitted.

If a ring is larger than 20 descriptors, there still should be one invalid entry in that ring. This is due to the fact that the ring may still wrap unexpectedly because there can be a delay in writing indications back to the ring if the Port output is busy. The indication not being written out does not prevent the FSI from reading more descriptors. This applies only to a descriptor ring that is smaller than the internal memory (8Kb or 1K descriptors).

APPENDIX E
ERROR DISCUSSION AND REFERENCE

E.1 OPERATIONAL ERRORS

E.1.1 Parity Treatment

The FSI supports parity generation and checking on the information transferred through it. When errors in parity occur, they are handled as indicated in the following sections.

E.1.1.1 Parity Checking

The source of the information written to a Port may be either the external memory or a host processor itself. All such information written to an FSI Port may be divided into the following categories:

a. Transmit Data, which goes through the Data Register (DTR),

b. Transmit and Receive Descriptors, which also goes through DTR,

c. Port directly accessed registers except the Data Register (DTR); i.e. SR1, IMR2, CMR, CER, FCR, IOR.

E.1.1.2 Transmit Data

Transmit data is written into one of the FSI's internal Transmit Data FIFOs according to its source: either into a Transmit Data FIFO or into the Command Data FIFO when Transmit Commands are given directly to the FSI. Since data may be written by the host processor directly or may be taken from an external memory, the user can select whether or not to check the data parity based on the Data Parity Enable (DPE) control bit inside an appropriate Ring Parameter Register (RPR) or Command Parameter Register (CPR). Even when external parity is not available and the parity check option is disabled, the MAC interface logic generates parity and appends it to the data for transfers between the FSI and the MAC. When a parity error is recognized by a Port during a Transmit Data transfer, the following actions are taken by the FSI:

a. Transfer of the frame into FSI internal memory is stopped immediately and the frame is marked as an error frame.

b. Transmission of the frame is aborted, possibly creating a remnant frame, if the frame was currently being transmitted.

c. The error indication for this frame is written back to the descriptor ring

Note that the activities on the Ring, (descriptor fetching, data reading, frame transmission and indication writing), will otherwise continue normally.

E.1.1.3 Buffer Descriptors

Descriptors are read by the FSI from a Descriptor Ring into one of the Internal Ring FIFOs. The decision as to whether or not to check data parity is based on the Ring's Ring Parity Enable (RPE) control bit inside the Ring Parameter Register (RPR). When a Parity Error is recognized by a Port during a descriptor transfer, the following actions are taken by the FSI:

a. The transfer is stopped immediately. The descriptor with a parity error will not be transferred to the internal Ring FIFO. The Ring Read Pointer, as accessed by the Read Ring Parameter command, will point to the beginning of this descriptor.

b. The state of this Ring is changed to Empty and the Parity Error status bit is set in the appropriate Ring State Register.

c. The Ring Not Ready (RNR) bit in the Status Register 1 (SR1) is set causing an interrupt if enabled.

d. The Ring Error (RER) bit in the Status Register 1 (SR1) is set causing an interrupt if enabled.

In many cases, a Parity Error can be caused by temporary problems on the external data bus, such as noise. Therefore, the host processor may choose to instruct the FSI to read this entry again. The following actions should be taken by the host processor to cause the FSI to re-read this entry:

a. clear the RER and RNR bits in the SR1 by writing the SR1 with these bits set,

b. transfer the ring to the Ready state by performing a Ring Ready control access.

If the parity error was a severe error, the Ring should be redefined to avoid this memory block. To do this, the host processor first issues the Stop Command to transfer the Ring to the Stopped state, and then redefines the Ring.

E.1.1.4 Directly Accessed Registers

Directly accessed registers are written by the host processor and, therefore, may or may not be covered by parity depending on the host processor bus. The Port Interface Logic checks parity if the Host Parity Enable (HPE) bit inside the Port Control Register (PCR) is set. When a parity error is detected on host processor accesses, the following actions are taken by the FSI:

a. If the access is a write to a status register, i.e., a reset, the reset does not take place. If the access is to a command register the command is not executed. If the access is to an FCR the internal control register is not read or written.

b. The Host Error (HER) bit in the Status Register 1 is set which causes an interrupt if enabled.

If the CER is written with a parity error, the command written to the CMR is ignored.

E.1.2 First or Last Bit Errors

The first descriptor of a frame should have the First bit set, and the last descriptor in a frame should have the Last bit set. Commands, and Transmit Commands when only one descriptor is used to define an entire frame, should have both the First and Last bits set. With these restrictions, the following error situations may occur:

a. The First descriptor might be detected when the Last descriptor of the previous frame is still expected.

b. The First bit may be zero when the first descriptor of a frame or a command is expected.

c. A command entry may have the Last bit set to zero.

In all of the above cases, the reaction of the FSI is as follows:

a. The transfer is immediately stopped. The entry which caused the error will not be transferred to the internal FSI Ring FIFO. The Ring Read Pointer will point to the beginning of this entry.

b. The Ring state is changed to Empty with the Operation Error status bit set in the appropriate Ring State Register.

c. The Ring Not Ready (RNR) and Ring Error (RER) bits in the General Status Register (SR1) are set causing an interrupt if enabled.

To recover from this error, the host processor should do the following:

a. Clear the RER and RNR bits in the SR1 by writing the SR1 with these bits set.

b. Change the error entry to a valid entry. The host processor can locate the error by reading the Ring Read Pointer by means of the Read Ring Parameters Command.

c. Transfer the Ring to the Ready state by performing a Ring Ready control access.

If the error is severe and the Ring should be redefined, the host processor should first issue the Stop Command to transfer the Ring to a Stopped state, and then redefine the Ring.

DMA Indication: Section 5.2.3.4

ER: Set when the DMA transfer is aborted due to an error specified by the PER bits

PER: 3 bit field specifying an error during data transfer through the FSI ports

DMA Error Indication: (destination side)

Section 5.2.3.6

PER: Specifies an error during data transfer through the FSI ports

IER Register: Section3.3.20

Also see section 3.2.4 IOE of the Status Register 1

IOE: if response time of Main Controller exceeds the maximum allowed limit, this will occur. Could indicate a problem with the Main Controller. Receiver is disabled (RE4 and 5 reset) and RABORT is generated.

IUE: If response time of Main controller exceeds maximum allowed limit for data transfer, this occurs. Could indicate a problem with the Main Controller. Frame being transmitted is aborted with the proper indication, and the transmitter is disabled (TE reset).

TPE: MAC interface recognizes parity error on transmit data. An internal parity error detected during data reads from the internal memory will always indicate a problem in an internal FSI data path. In this case the TPE bit is set , the frame currently being transmitted is aborted and the TE (transmit enable) bit in the MACIF transmit control register is reset.

MER: The Mac interface reciever recognizes a protocol handshake error between the MAC and FSI. The MACIF will generate an RABORT to the MAC and try to resynchronize with the MAC on the next frame. This can be caused by temporary problems such as noise on control lines so no special recovery functions are required. If there are permanent problems in the interface between the FSI and MAC the Host processor should perform an external diagnostic.

MOV: FSI experiences an internal memory overrun causing the MOV to be set. This can be caused by an incorrect definition of an internal FIFO's watermark. Check FSI parameter definitions and reset the FSI.

PBE/PAE: Port interface recognizes a parity error on received data being transferred to the system bus.

Overrun/Underrun:

Overrun occurs if FSI receiver is unable to move data from its receive buffers to system memory in adequate time to prevent receive buffer overflow. The receiver is accumulating data sent from the MAC in much less time than it takes the FSI to aquire the system bus.

Underrun occurs if the FSI transmitter is unable to move data from system memory to it's transmit buffers in adequate time to prevent transmit buffer underflow. The transmitter is sending data to the MAC in much less time than the FSI is receiving data from the system bus.

Parity Errors:

Bits Affected by Parity Errors:

Bit:	Register/Indication:	Location of error:
TPE	IER	MACIF sees parity error on tx data
PAE	IER	Port interface sees parity error on rx data
PBE	IER	being transferred to System Bus
HER	SR1	Parity error on write to direct access register
DRE	SR2	Parity error on access to destination ring entry
PE	Rx Error Indic.	Parity error detected by FSI
PER	RSR	Parity detected on Ring entry Read
PER	Tx Indic.	PER=010, 110, 111 (pg 47 FSI3.2)
PE	Tx Indic.	
PER	DMA Indc.	PER=010,111 (pg 48, FSI3.2)

Parity Error Handling

Transmit data parity error: Section 7.3.1.2

When there is a parity error recognized by a port during a Transmit data transfer the FSI

a) Immediately stops the transfer of the frame to FSI internal memory and the frame is marked as an error frame

b) Aborts transmission of the frame, possibly creating a remnant frame if the frame is currently being transmitted

c) The error indication for this frame is written back to the descriptor ring

Note that the activities on the ring will otherwise continue normally

Buffer Descriptor transfer parity error: Section 7.3.1.3

When a parity error is recognized by a port during a descriptor transfer the FSI

a) Immediately stops the transfer. The descriptor will not be transferred to the internal Ring FIFO. The Ring Read Pointer will point to the beginning of this descriptor as accessed by the Read Ring Parameter command.

b) The state of this Ring is changed to Empty and the Parity Error status bit is set in the appropriate Ring State Register

c) The Ring Not Ready bit in the Status Register 1 is set causing an interrupt if enabled

d) The Ring Error bit in the Status Register 1 is set causing an interrupt if enabled

To cause the FSI to re-read this entry in the case of a parity error being caused by temporary problems on the data bus the host processor should

a) clear the RER and RNR bits in the SR1 by writing the SR1 with these bits set

b) transfer the ring to the ready state by performing a Ring Ready control access

If the parity error was a severe error the Ring should be re-defined to avoid this memory block. To do this the host processor first issues the Stop command to transfer the Ring to the Stopped state and then re-defines the ring.

Host processor access to Direct Access register parity error: Section 7.3.1.4

When a parity error is detected on host processor accesses the FSI performs the following:

a) the host processor access is ignored

b) transfer the ring to Ready state by performing a Ring Ready control access.

If the CER is written with a parity error, the command written to the CMR is ignored.

Port Operation Errors:

Discussed thoroughly in section 8.4.3

Ring State Register: Section 3.3.6

PER: set when FSI detects a parity error while a Ring entry is being read. It is reset when

a) the ring is re-defined

b) Ring Ready is signaled (by control access to FCR)

OER: set when a POE is recieved or First/Last bit error

For discussion on First/Last bit errors, see section 7.3.2

It is reset when

a) the ring is re-defined

b) Ring Ready is signaled (by control access to FCR)

Status Register 1: Section 3.2.4

IOE: Internal Operation Error detected. This is a sticky bit an must be cleared by the host. Internal Operation errors relate to the IER register discussed above and in Section 3.3.20.

ROV4-5: An overrun condition has occured for this ring. This bit is sticky and must be cleared by a host write to the Status Register 1.

RER: Set by :

a) Parity Error

b) Port Operation Error

c) First or Last bit Error

Status Register 2: Section 3.2.5

DRE: set by Parity Error or Port Operation Error

Receive Frame Normal Indication: Section 5.2.3.10

CE : CRC error. The Cyclical Redundancy check on the frame does not agree/pass. There are no guarentees that the frame is good if this bit is set.

Receive Frame Error Indication: Section 5.2.3.11

Generated when

a) error, fragment or secondary NSA frame is received when FSI is in Receive All mode

b) the receive operation has been aborted by the FSI

c) the fragment is longer than the receive watermark.

STT: indicates the receive status of the MAC

Transmit Indication: Section 5.2.3.2

PER: Specifies an error that occured during data transfer through the FSI ports

AB: Indicates that frame transmission has been aborted

UN: Indicates that an underrun error caused a frame to be aborted. Following transmit frames will occur normally. Only the frame which is delimited by the descriptors when the underrun occured is discarded.

PE: Indicates that transmission of the frame was aborted due to the occurence of a parity error.

-> if AB=1, UN=PE=0 implies that the decision to abort frame transmission is due to assertion of the TABORT signal.

MOTOROLA

Literature Distribution Centers:
USA: Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036.
EUROPE: Motorola Ltd.; European Literature Center; 88Tanners Drive, Blakelands, Milton Keynes, MK14 5BP, England.
JAPAN: Nippon Motorola Ltd.; 4-32-1,Nishi-Gotanda, Shinagawa-ku, Tokyo 141 Japan.
ASIA-PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Center, No. 2 Dai King Street, Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong.