SX1601
IPI-2 DISK DRIVE
INTERFACE PROTOCOL CIRCUIT

Features

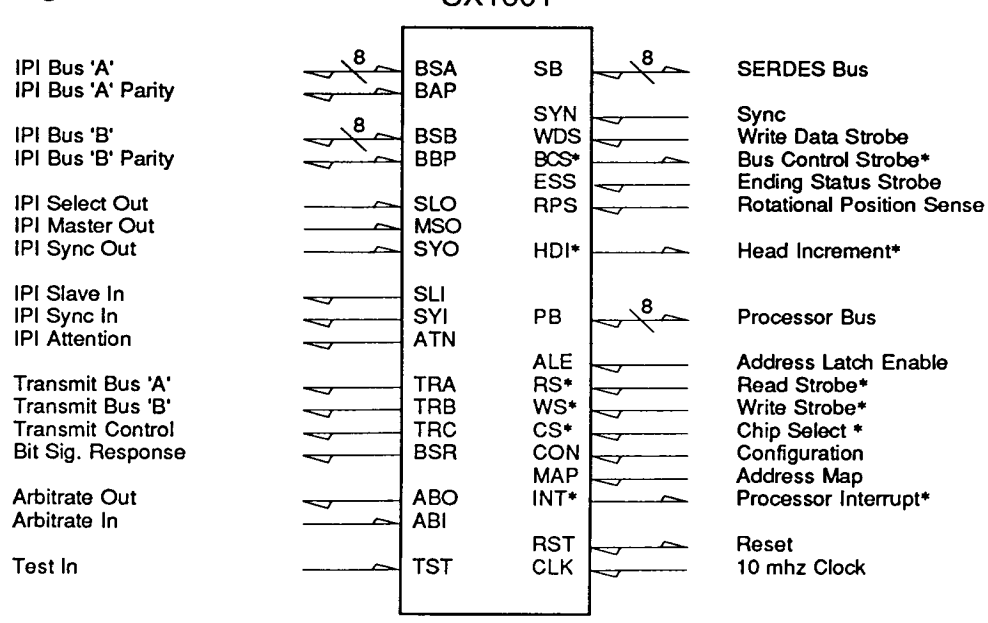
- Generates and Checks IPI Bus parity
- Internal FIFO buffers disk write data
- Supports IPI Maintenance Mode 1
- Supports Dual Port drive configurations
- Supports disk transfer rates up to 40 Mhz
- 68 pin PLCC package
- Directs Data Bus Controls to the SERDES / FORMATTER CIRCUIT
- Directs Command / Response Bus Controls to the Drive Processor
- Performs the following sequences without Drive Processor intervention:
  - Selection Sequence
  - Request Interrupts Sequence
  - Request Slave Interrupts Sequence
  - Request Transfer Setting Sequence
  - Selective Reset Sequence

Description

The IPI-2 Interface Protocol Circuit (IPC) is a 68-pin semi-custom CMOS VLSI integrated circuit. The IPC circuit is intended to provide the IPI Bus Interface functions for a magnetic disk drive implementing the IPI-2 interface. The IPC, in conjunction with the Serdes/Formatter Circuit, interprets all IPI Bus Sequences. Command / Response Bus Controls are directed to the drive processor; Data Bus Controls are directed to the Serdes / Formatter Circuit.

In dual port implementations, two Interface Protocol Circuits are employed. The two Circuits arbitrate for control of the drive. The IPC supports arbitration for selection, or arbitration for control of drive HDA functions.

Logic Diagram



Rev. 4.1 1/16/87

1.0 APPLICABLE DOCUMENTS

The following documents are considered a part of this specification:

1) Simulex Corporation Product Specification, SX1602 Serdes/Formatter Circuit (SFC)
2) ANSI Standard X3.129, IPI Physical
3) ANSI Standard X3.130, IPI Command Set, Device Specific for Magnetic Disk

1.1 BLOCK DIAGRAM

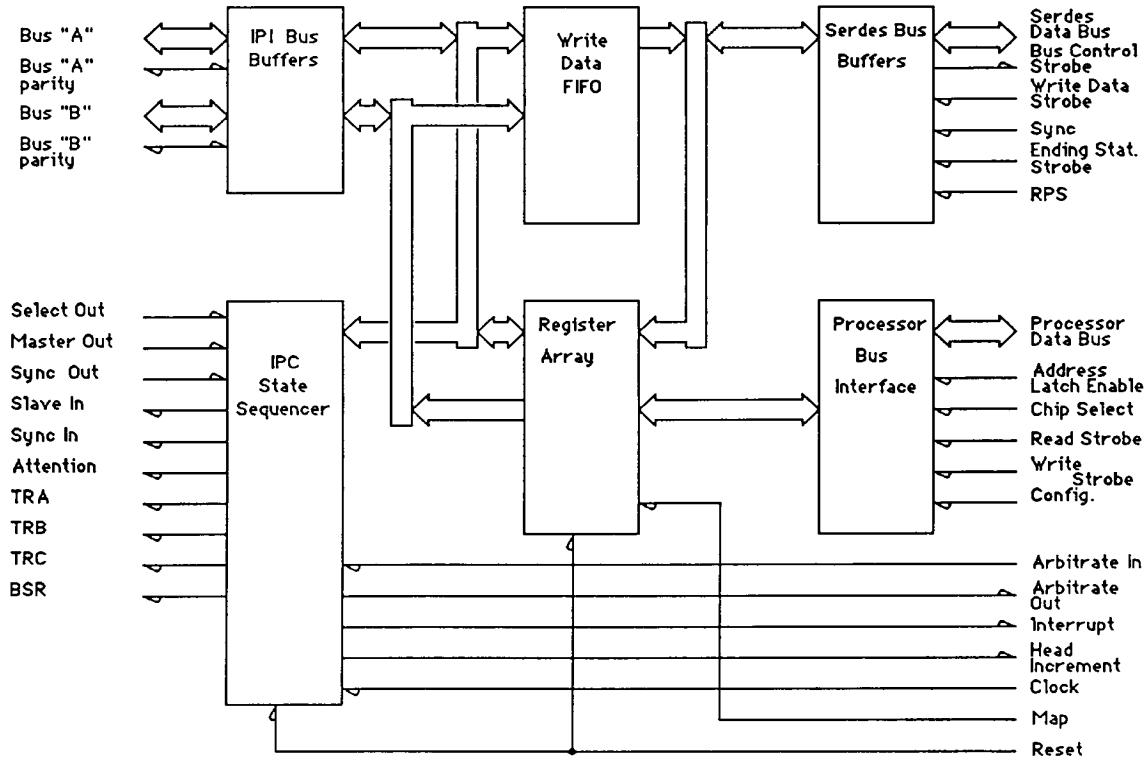


Figure 1
Interface Protocol Circuit
Block Diagram

2.0 SIGNAL DESCRIPTIONS

The signal description table provides a brief description of the function of each signal and its mnemonic. Signal mnemonics ending in an " * " are active low, all other signals are active high.

2.1 PROCESSOR INTERFACE SIGNALS

Signal Mnemonic	Input/ Output	Signal Description
PB 0-7	I/O	**Processor Address/Data Bus** The Processor Bus lines are TTL compatible bi-directional 3-state active high signals for data exchange between the drive processor and the IPC. The bus address is captured at the beginning of a read or write cycle when Address Latch Enable goes inactive. The outputs are enabled during a read cycle to allow the contents of the IPC internal registers to be examined by the drive processor. The outputs are disabled and the internal inputs are enabled during a write cycle to allow the contents of the internal registers to be altered.
RS*(DS*)	I	**Read Strobe (Data Strobe)** This TTL compatible high impedance active low input performs one of two functions depending on the state of the CON input. When CON is inactive, this input becomes a Read Strobe signal and enables the contents of the addressed register to be output to the Processor Bus while the RS* signal is active. When CON is active, this input becomes a Data Strobe and enables the contents of the Processor Bus to be latched into the addressed register if R/W is active, or allows the contents of the addressed to be output to the Processor Bus if R/W is inactive.
WS*(R/W)	I	**Write Strobe (Read/Write)** This TTL compatible high impedance active low input performs one of two functions depending on the state of the CON input. When CON is inactive, this input becomes a Write Strobe and causes the contents of the Processor Bus to be latched by the addressed register on the active to inactive going edge of the WS* signal. When CON is active, this input becomes a Read/Write signal and determines whether a Processor Bus read or write cycle will be performed when DS* is active. If R/W is active, the active going edge of DS* starts are read cycle. If R/W is inactive, the active going edge of DS* starts are write cycle.
CS*	I	**Chip Select** This TTL compatible high impedance active low input enables the IPC to respond to the RS and WS signals.

ALE	**I**	**Address Latch Enable** This TTL compatible high impedance input indicates when a valid address is present on the Processor Bus. The address is internally latched on the active to inactive going edge of the ALE signal. When CON is inactive this input is active high. When CON is active this input is active low.
INT*	**O**	**Interrupt** This TTL compatible open drain active low output is asserted to interrupt the drive processor when the Interface Protocol Circuit requires service.
MAP	**I**	**Map** This TTL compatible high impedance active high input determines the location of the IPC internal register within the address map. When inactive, the register array appears in the first 16 memory locations of the address map. When active, the register array appears in the second 16 memory locations.
RST	**I/O**	**Reset** This TTL compatible bi-directional signal resets and initializes the IPC circuit when an active level input is detected for five or more clock cycles. This signal is driven active by the Interface Protocol Circuit for 12.8 microsecondes after the receipt of a Selective Reset Octet with Bit 2 set. When the CON signal is high, the RST signal is active low. When the CON signal is low, the RST signal is active high.
CON	**I**	**Configuration** This TTL compatible high impedance active high input determines the processor interface mode of operation. When low, the processor bus control signals, are configured as RD* and WR* and behave in a manner compatiable with the Intel 8051. When this signal is high, the processor bus control signals, are configured as R/W and DS* and behave in a manner compatiable with the Zilog Z8 and Motorola 6801 processors.
CLK	**I**	**10 Megahertz Clock** This TTL compatible high impedance active high signal is the circuit clock. Frequency must be 10 MHz ±1%. Symmetry must be 50/50 ±5%.

2.2 SERDES BUS INTERFACE SIGNALS

SB 0-7	**I/O**	**Serdes Data Bus** The Serdes Data Bus lines are CMOS bi-directional 3-state active high signals for data exchange between the Serdes/Formatter Circuit and the IPC. Bus Controls, read/write disk data, and an Ending Status Byte are passed over this bus for each Data Control received by the IPC.
BCS*	**O**	**Bus Control Strobe** This CMOS active low 3-state output signals the receipt by the IPC of a Data Bus Control from the IPI interface. The active going edge of this signal indicates that a Data Bus Control is being placed on the Serdes Bus data lines. This signal goes inactive when a Master Termination of the data transfer is recognized by the IPC, or when the SFC initiates a Slave Termination of the data transfer by asserting Ending Status Strobe.
WDS	**I**	**Write Data Strobe** This CMOS high impedance active high input is used during disk write operations to indicate when disk write data is to be place on the Serdes Bus data lines from the IPC data FIFO. The FIFO is advanced and the next byte of write data is placed on the Serdes Bus on the inactive going edge of this signal. During disk read operations the Write Data Strobe signal is inactive.
SYN	**I**	**Sync** This CMOS high impedance active high input receives a pulse stream from SFC during the execution of disk read/write Data Controls. This pulse stream is used by the IPC to generate IPI Sync In pulses. During the execution of disk read Data Controls this signal also indicates the presence of valid disk data on the Serdes Bus.
ESS	**I**	**Ending Status Strobe** This CMOS high impedance active high input indicates the end of a disk read/write Data Control operation, and the presences of an Ending Status Octet on the Serdes Bus data lines. When this signal goes active, the IPC is required to drive BCS to the inactive state if it not in that condition, and to release the BCS signal and Serdes Bus data lines in preparation for the transfer of ending status. When this signal goes inactive, an Ending Status Octet is being placed on the Serdes Bus data lines by the SFC.
RPS	**I**	**Rotation Position Sensing** This CMOS high impedance active high input, when active, indicates that the disk data heads are over the target area.

2.3 IPI BUS SIGNALS

BA 0-7	**I/O**	**Bus "A"** These TTL compatible bi-directional 3-state active high signals are equivalent to the IPI Bus "A" lines over which data is transferred on the IPI Bus. Bus "A" data is assumed to logically arrive before Bus "B" data.
BAP	**I/O**	**Bus "A" Parity** This TTL compatible bi-directional active high signal transmits or receives parity associated with the IPI Bus "A".
BB 0-7	**I/O**	**Bus "B"** These TTL compatible bi-directional active high signals are equivalent to the IPI Bus "B" lines over which data is transferred on the IPI Bus. Bus "B" data is assumed to logically arrive after Bus "A" data.
BBP	**I/O**	**Bus "B" Parity** This TTL compatible bi-directional active high signal transmits or receives parity associated with the IPI "B" Bus.
SLO	**I**	**Select Out** This TTL compatible high impedance active high input is equivalent to the IPI Select Out signal.
MSO	**I**	**Master Out** This TTL compatible high impedance active high input is equivalent to the IPI Master Out signal.
SYO	**I**	**Sync Out** This TTL compatible high impedance active high input is equivalent to the IPI Sync Out signal.
SLI	**O**	**Slave In** This TTL compatible active high output is equivalent to the IPI Slave In signal.
SYI	**O**	**Sync In** This TTL compatible active high output is equivalent to the IPI Sync In signal.
ATN	**O**	**Attention** This TTL compatible active high output is equivalent to the IPI Attention signal.
TRA	**O**	**Transmit/Receive "A" Bus** This TTL compatible active high output, when active, indicates that the IPC is outputting data to the IPI "A" Bus lines. This signal is provided to allow for the control of the direction of the IPI "A" Bus transmitter/ receivers.

TRB	**O**	**Transmit/Receive "B" Bus** This TTL compatible active high output, when active, indicates that the IPC is outputting data to the IPI "B" Bus lines. This signal is provided to allow for the control of the direction of the IPI "B" Bus transmitter/ receivers.
TRC	**O**	**Transmit Control Signals** This TTL compatible active high output, when active, indicates that the IPC has been selected. This signal is used to enable the Slave In and Sync In transmitter/receivers.
BSR	**O**	**Bit Significant Response** This TTL compatible active high output, when active, indicates that the Bus "B" lines contain a Bit Significant Response to a selection sequence. This signal is provided to allow for the control of the BSR multiplexer.
ABI	**I**	**Arbitrate In** This CMOS high impedance active high input, when active, indicates a request by the alternate port IPC for control of the drive or the drive HDA, depending on the Arbitration mode set in the Control register. If this signal is active the IPC may not assert Arbitrate Out for the purpose of gaining control of the same resource.
ABO	**O**	**Arbitrate Out** This CMOS active high output, when active, indicates a request by the IPC to the alternate port IPC for control of the drive or control of the HDA, depending on the Arbitration mode set in the Control register. If after asserting Arbitrate Out, the Arbitrate In signal remains inactive then the IPC has gained control. If not, then the IPC must negate Arbitrate Out and reject the Port operation.

2.4 DISK INTERFACE

HDI*	O	**Head Increment** This TTL compatible active low open drain output is used to advance the drive head address counter after the successful completion of a Data Bus Control with the Head Increment bit set.

2.5 DEVICE TEST

TST	I	**Test Input** This input is used during manufacturing for input threshold testing. It is grounded for normal operation.

2.6 POWER

VCC1	Positive Five Volt Power
VCC2	Positive Five Volt Power
GND1	Power Ground and Signal Reference
GND2	Power Ground and Signal Reference
GND3	Power Ground and Signal Reference
GND4	Power Ground and Signal Reference

2.7 SIGNAL CHARACTERISTICS

Signal Mnemonic	Input/ Output	Output I_{OL} (ma)	Output I_{OH} (ma)	Input Threshold	Input Setup (ns)	Input Hold (ns)
MAP	INPUT			TTL	Note 1	
RS*	INPUT			TTL	Note 1	
WS*	INPUT			TTL	Note 1	
CS*	INPUT			TTL	10.0	10.0
ALE	INPUT			TTL	Note 1	
CON	INPUT			TTL	Note 1	
CLK	INPUT			TTL	Note 1	
SLO	INPUT			TTL	Note 1	
MSO	INPUT			TTL	Note 1	
SYO	INPUT			TTL	Note 1	
ABI	INPUT			CMOS	15.0	5.0
WDS	INPUT			CMOS	Note 1	
SYN	INPUT			CMOS	Note 1	
ESS	INPUT			CMOS	15.0	5.0
RPS	INPUT			CMOS	Note 1	
TST	INPUT			TTL	Note 1	
SLI	OUTPUT	4.0	4.0			
SYI	OUTPUT	4.0	4.0			
ATN	OUTPUT	4.0	4.0			
ABO	OUTPUT	4.0	4.0			
TRA	OUTPUT	4.0	4.0			
TRB	OUTPUT	4.0	4.0			
TRC	OUTPUT	4.0	4.0			
BSR	OUTPUT	4.0	4.0			
BCS*	OUTPUT	4.0	4.0			
HDI*	OUTPUT	12.0	(open drain)			
INT*	OUTPUT	12.0	(open drain)			
BA(0-P)	I/O	4.0	4.0	TTL		
BB(0-P)	I/O	4.0	4.0	TTL		
SB(0-7)	I/O	4.0	4.0	CMOS		
PB(0-7)	I/O	12.0	12.0	TTL		
RST	I/O	12.0	12.0	TTL		

Note 1 - No Setup or Hold requirements.

2.8 SIGNAL PIN ASSIGNMENTS

SIGNAL	PIN	SIGNAL	PIN
BA0	27	BA1	26
BA2	25	BA3	24
BA4	23	BA5	22
BA6	21	BA7	20
BAP	19	BB0	38
BB1	37	BB2	34
BB3	33	BB4	32
BB5	31	BB6	30
BB7	29	BBP	28
SB0	66	SB1	67
SB2	2	SB3	3
SB4	4	SB5	5
SB6	6	SB7	7
PB0	50	PB1	51
PB2	52	PB3	54
PB4	55	PB5	56
PB6	57	PB7	58
SLO	46	MSO	45
SYO	44	SLI	14
SYI	15	ATN	16
ABI	43	ABO	17
TRA	42	TRB	41
TRC	40	WDS	12
SYN	11	ESS	10
RPS	9	BCS*	8
HDI*	13	MAP	61
RS*	62	WS*	60
CS*	59	ALE	49
CLK	48	INT*	39
RST	65	TST	47
BSR	64	CON	63
VCC1	36	VCC2	68
GND1	1	GND2	18
GND3	35	GND4	53

2.9 PACKAGE PIN ASSIGNMENTS

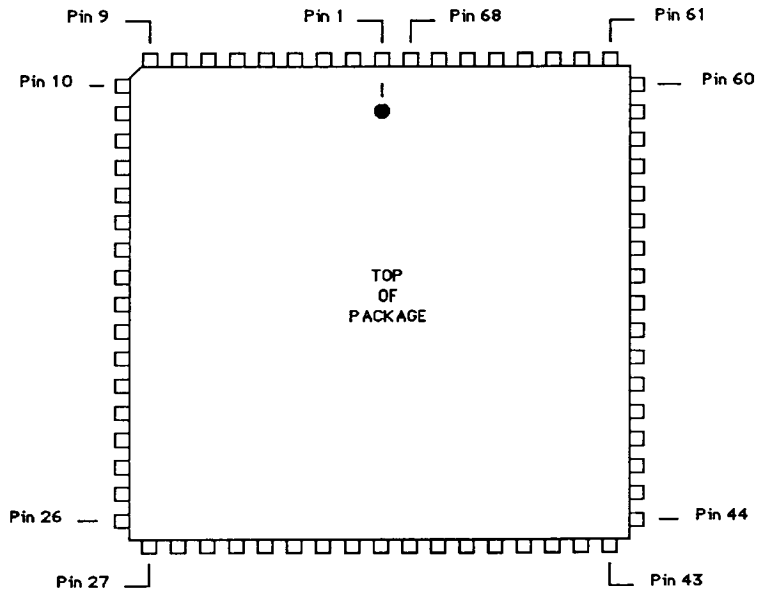


Figure 2
68 Pin PLCC
Pin Assignments

3.0 FUNCTIONAL DESCRIPTION

The Intelligent Peripheral Interface, Level-2, specifies a bit-parallel word-serial bus structure intended for communication between a Host Controller and up to eight magnetic disk drives.

The Interface Protocol Circuit (IPC) is a single-chip implementation of the interface circuitry for an IPI level-2 disk drive. The IPC interprets all IPI state sequences. Command/Response Bus Controls are passed to the drive processor for execution, and Data Bus Controls are passed to the Serdes/Formatter Circuit for execution.

There are six major components to the circuit: the Processor Bus Interface, Register Array, Serdes Bus Interface, IPI Bus Buffers, State Sequencer, and Disk Data FIFO. The following sections describe the various components of the circuit.

3.1 Processor Bus Interface

The Processor Bus Interface, in conjunction with the Register Array, provides the primary means for controlling and directing the operation of the Drive Interface Circuit. The drive processor configures and programs the IPC mode of operation through the values stored in the Register Array.

The IPC acts as a processor bus slave. The Processor Bus consists of an eight-bit address/data bus, and the Address Latch Enable, Chip Select, and two bus control signals. The function of the bus control signals is determined by the state of the IPC Configuration signal (CON). If CON is inactive, the two bus control signals function as Read Strobe and Write Strobe, in a manner compatible with the Intel 8051 family of micro-controllers. If CON is active, the two bus control signals function as Read/Write and Data Strobe, in a manner compatible with the Zilog Z8 and Motorola 6801 micro-controllers.

The active state of ALE indicates that a valid address is on the data bus. The IPC latches this address and the state of the Chip Select signal in an internal address register. If the Chip Select signal was active and the address falls within the address map of the IPC, the circuit becomes selected.

When the CON signal is inactive: If Read Strobe goes active and the circuit is selected, the contents of the addressed register are output to the data bus lines. If Write Strobe goes active and the circuit is selected, the eight-bit data value present on the data bus is stored in the addressed register.

When the CON signal is active: If Data Strobe goes active when Read/Write is true and the circuit is selected, the contents of the addressed register are output to the data bus lines. If Data Strobe goes active when Read/Write is not true and the circuit is selected, the eight-bit data value present on the data bus is stored in the addressed register.

The Processor Bus Interface is directly timing-compatible with the Intel 8051, Zilog Z8, or Motorola 6801 family of microprocessors. With the appropriate logic to multiplex the address onto the data bus and to generate the proper bus control signals, most eight-bit microprocessors can be adapted for use with the SFC.

3.2 Register Array

The register array consists of 16 register addresses (the first 8 are used, the second 8 are reserved), as described below. The address of the registers within the 256 byte address space depends on the state of the MAP input signal. The purpose of the MAP signal is to allow two Interface Protocol Circuits in a dual port disk drive to connect to the same Processor Bus without addressing conflict. If the MAP signal is inactive, the register map lies in the first 16 locations of the address space: 00-0F hex. If the MAP signal is active, the register map lies in the second 16 locations of the address space: 10-1F hex.

All registers can be read or written by the drive processor unless otherwise specified. Table 1 lists the registers and their hexa-decimal addresses.

Table 1 List of Registers

Address Low Map	Address High Map	Description
00	10	Control Register
01	11	Status Register
02	12	MPU Interrupt Register
03	13	Configuration Register
04	14	Slave Interrupts Register
05	15	Bus "A" Register
06	16	Bus "B" Register
07	17	Bus Control/Slave Status Register
08	18	Not used (Reserved)
09	19	Not used (Reserved)
0A	1A	Not used (Reserved)
0B	1B	Not used (Reserved)
0C	1C	Not used (Reserved)
0D	1D	Not used (Reserved)
0E	1E	Not used (Reserved)
0F	1F	Not used (Reserved)

A description of each addressable register follows.

3.2.1 Control Register (00)

This eight-bit read/write register controls the operation of the Interface Protocol Circuit. The Circuit can be enabled, forced to appear busy, forced to reject certain Bus Controls, or put into the Diagnostics mode through the control bits contained in this register. When read, all bits return the last value written. Changes in bits that control the operation of the port on the interface will not take effect unless, or until, the port is in state 0 (interface at IDLE) or state 2 (port selected and interface at BUSACK) as defined below. All control bits are reset by the external Reset input. The definition of each bit is outlined below:

Bit	Description
7	Enable Port This bit, in conjunction with Control Register Bit 0, controls the enabling or disabling of the port. See Control Register Bit 0 for a full description.
6	Force Selection Busy This bit, when set, forces the Interface Protocol Circuit to respond as busy (SLI asserted with no Bit Significant Response) during a selection sequence. This bit will only take effect when the port is in, or after it passes through, state 0 or 2.
5	Force Command/Response Busy This bit, when set, forces a drive busy Slave Status in response to any Command/Response Bus Control. This bit will only take effect when the port is in, or after it passes through, state 0 or 2.

4	Force Data Control Busy This bit, when set, forces a drive busy Slave Status in response to any Data Bus Control. This bit will only take effect when the port is in, or after it passes through, state 0 or 2.
3	Force Write Protect This bit, when set, forces an Operation Exception Slave Status in response to any Write Data Bus Control. If a Write Data Bus Control is received and rejected with Write Protected Slave Status an interrupt is set to notify the microprocessor (Write Protected status can then be logged for a subsequent Read Status Response). This bit will only take effect when the port is in, or after it passes through, state 0 or 2.
2	Enable Scan String Diagnostics This bit, when set, enables Scan String Diagnostics: all flip-flops in the State Sequencer are logically connected to form a shift register. The shift register is clocked via Control Register Bit 1. The shift register data input is Control Register Bit 0, and the data output is Status Register Bit 0. This bit, when reset, disables Scan String Diagnostics and Control Register Bit 1 and 0 are defined as Force Unsolicited Exception and Priority Select respectivly.
1	When Control Register Bit 2 is a one: this bit, when set from zero to one, clocks the value of Control Register Bit 0 into the Scan String shift register. Force Unsolicited Exception/Scan String Diagnostic Clock When Control Register Bit 2 is a zero: this bit, when set , forces Unsolicited Exception Slave Status in response to any Bus Control received. This bit will only take effect when the port is in, or after it passes through, state 0 or 2.
0	Priority Select/Scan String Diagnostic Data When Control Register Bit 2 is a one: this bit represents the value to be clocked into the Scan String Diagnostics shift register when Bit 1 transitions from a zero to a one. The sequence of the flip-flops in the Scan String (first bit clocked in to last bit clocked out) are shown in Table 2 below.

Table 2
Scan String Sequence

1)	STATE F LATCHED
2)	STATE F
3)	STATE E
4)	STATE D
5)	STATE C
6)	STATE B
7)	STATE 9 LATCHED
8)	STATE 9
9)	STATE 8 LATCHED
10)	STATE 8
11)	STATE 7 LATCHED
12)	STATE 7
13)	STATE 6
14)	STATE 5 LATCHED
15)	STATE 5 LATCHED
16)	STATE 4 LATCHED
17)	STATE 4
18)	STATE 3 LATCHED DELAYED
19)	STATE 3 LATCHED
20)	STATE 3
21)	STATE 2 LATCHED DELAYED
22)	STATE 2 LATCHED
23)	STATE 2
24)	STATE 1 LATCHED
25)	STATE 1
26)	STATE 0 LATCHED
27)	STATE 0

When Control Register Bit 2 is a zero: this bit, in conjunction with Control Register Bit 7, determines the enable/disable state of the port and the state of the Priority Select bit in the Slave Interrupt octet. The action taken when the Control Register is written into is described below:

Bit 7	Bit 0	Action Taken
0	0	Disable the port as soon as it is in state 0 or 2. The Priority Select bit in the Slave Interrupts Octet is not altered.
0	1	Disable the port as soon as it is in state 0 or 2. The Priority Select bit in the Slave Interrupts Octet is set to a one. The Priority Select bit will be reset after it is presented during a Slave Interrupts sequence or the next time the port is successfully selected.
1	0	Enable the port as soon as it is in state 0. The Priority Select bit in the Slave Interrupts Octet is not altered.

1	1	Disable the port immediately (suppressing any invalid sequence indications) and terminate any Data Bus Control in progress. Re-enable when the port goes to state 0. Set the Priority Select bit in the Slave Interrupts Octet to a one. The Priority Select bit will be reset after it is presented during a Slave Interrupts sequence or the next time the port is successfully selected.

Note: The current state of the port enable latch and the Priority Select bit can be read in the Status Register.

3.2.2 Status Register (01)

This eight-bit register contains the current state of the Interface Protocol Circuit. Bit 3 of this register is read/write, all other bits are read only. Through this register, the drive processor can determine whether Command/Response parameters are available/required, whether Master Status is available, whether a parity error or invalid state were detected, the current state of the port enable latch and the Priority Select bit, or the state of the Scan String Diagnostics output flip-flop. All status bits are reset by the external Reset input. The definition of each bit is outlined below:

Bit	Description
7	Parity Error This bit is set when the information read from BUS A or BUS B and stored in the, Bus "A" or Bus "B" registers, during a command transfer sequence, contains a parity error. This bit is cleared after the Slave Status octet is transferred to the Master during the Ending Status Sequence.
6	Data Ready/Request This bit is set when data is ready, or required, in the Bus "A" or Bus "B" register, during a command/response transfer sequence. This bit is cleared by writing into the Bus "B" Register (a dummy write is required after reading Bus "A" and Bus "B" during a command transfer) or when entering the Ending Status sequence.
5	Master Status Ready This bit is set when Master Status has been received in the Bus "A" register, and is cleared after the Slave Status octet is transferred to the Master during the Ending Status Sequence.
4	Invalid State This bit reflects the current value of the Invalid State interrupt latch.
3	Data Control Issued This bit is set when a Data Bus Control is received and is reset by writing a one into this location.
2	Port Enabled This bit reflects the current value of the port enable latch.
1	Priority Select Interrupt Status

	This bit reflects the current value of the Priority Select bit in the Slave Interrupts octet.
0	Scan String Diagnostic Data Out This bit contains the output value of the last flip-flop in the Scan String shift register.

3.2.3 MPU Interrupt Register (02)

This eight-bit register contains the current interrupt state of the Interface Protocol Circuit. Bits 6 and 7 are read only, all other bits are read/write. The INT signal is the logical OR of all the bits in this register. The drive processor can determine through this register the cause of a processor interrupt; Bus Control Available, Power Fail Alert, Logical Reset, Illegal State Transition, Write Protected or Priority Reserve. The processor clears Interrupts (except Bus Control Available and Power Fail Alert) by writing a one back to its bit position in the Interrupt Register. All interrupt bits are reset by the external Reset input. The definition of each bit is outlined below:

Bit	Description
7	Bus Control Available Interrupt This bit is set when a Command/Response Bus Control is received, and is cleared when the Bus Control Register is read.
6	Power Fail Alert Interrupt This Bit is set when a Request Interrupt Octet is received with the Power Fail Alert Bit set, and is cleared by setting the Power Fail Response Bit in the Slave Interrupts Register.
5	Logical Reset Interrupt This bit is set when a Selective Reset Octet is received with the Logical Reset Bit asserted, and is cleared by writing a one to this bit location.
4	Invalid State Sequence Interrupt This bit is set when an illegal state transition is observed by the IPC, and is cleared by writing a one to this bit location.
3	Write Protected Interrupt This bit is set when a write type Data Bus Control is rejected because the Force Write Protect Bit is set in the Control Register. This bit is cleared by writing a one to this location.
2	Not used, zero
1	Not used, zero
0	Priority Reserve Interrupt This bit is set when a Selection octet is received with the Priority Reserve bit asserted. This bit is cleared by writing a one to this location.

3.2.4 Configuration Register (03)

This eight-bit read/write register contains the current interrupt mask settings, the port address, port Arbitration Mode and the enable for No Longer Busy Attention. The definition of each bit is outlined below:

Bit	Description
7	No Longer Busy Attention Enable This bit, when set, allows a No Longer Busy condition to assert the IPI Attention signal.
6-4	Port Address This 3-bit field contains the IPC port address. Bit 6 is the MSB.
3	Arbitration Mode This bit governs the usage of the IPC arbitration signals, Arbitrate In and Arbitrate Out, in a dual port drive application. When this bit is cleared in both Interface Protocol Circuits of a dual port drive, the ABI and ABO signals are used to arbitrate selection between the drive ports. Only one port may gain selection at any given time. When this bit is set to a one in both Interface Protocol Circuits of a dual port drive, the ABI and ABO signals are used to arbitrate for access to the Serdes/ Formatter Circuit when a Bus Control is received. This bit will only take effect when the port is in, or after it passes through, state 0 or 2.
2	Enable Class 3 Interrupt This bit, when set, enables the Class 3 Interrupt Bit (Request Interrupts Register bit 2) to assert the IPI Attention signal.
1	Enable Class 2 Interrupt This bit, when set, enables the Class 2 Interrupt Bit (RPS signal from the SFC) to assert the IPI Attention signal.
0	Enable Class 1 Interrupt This bit, when set, enables the Class 1 Interrupt Bit (Request Interrupts Register bit 0) to assert the IPI Attention signal.

3.2.5 Slave Interrupts Register (04)

This eight-bit read/write register contains the information sent to the Master over Bus "B" during a Request Slave Interrupts sequence. It is also compared against the Request Interrupts Octet during an IPI Request Interrupts sequence to determine if a Bit Significant Response is to be generated. Bit 7 is used to for port control and is not part of the Slave Interrupts. The definition of each bit is outlined below:

Bit	Description
7	Switched This bit, when set, indicates that the microprocessor has "switched" control to the other port and supresses the Attention signal on this port. When read it returns the value last written.
6	Busy Status This bit is set by the drive processor to indicate a drive busy condition. When read it returns the value last written.
5	Ready Status This bit is set by the drive processor to indicate a drive ready condition. When read it returns the value last written.
4	Power Fail Response This bit is set by the drive processor to acknowledge the receipt of a

	Power Fail Alert from the controller. When read it returns the value last written.
3	Power On Status This bit is set by the drive processor to indicate a drive in operation condition. When read it returns the value last written.
2	Class 3 Interrupt (Status Pending Interrupt) This bit is set by the drive processor to indicate a Class 3 interrupt condition. This bit is ANDed with the Enable Class 3 Interrupt Bit (Interrupt Mask Register bit 2) to generate the IPI Attention Signal. When read it returns the value last written.
1	Class 2 Interrupt (RPS) This read only bit indicates the current state of the RPS interrupt signal.
0	Class 1 Interrupt (Command Completion Interrupt) This bit is set by the drive processor to indicate a Class 1 interrupt condition. This bit is ANDed with the Enable Class 1 Interrupt Bit (Interrupt Mask Register bit 0) to generate the IPI Attention Signal. This bit is cleared by the receipt of any Bus Control.

3.2.6 Bus "A" Register (05)

This eight bit read/write register is used to receive Master Status Octets, the high order byte of command parameters, or is available for the transfer of the high order byte of a response parameter. Master Status Octets, and the high order byte of command parameters are obtained from the FIFO when this register is read. The high order byte of response parameters are stored in the BUS "A" latch when this register is written.

3.2.7 Bus "B" Register (06)

This eight bit read/write register is used to receive the low order byte of command parameters, or is available for the transfer of the low order byte of response parameters. The low order byte of command parameters are obtained from the FIFO when this register is read. The low order byte of response parameters are store in the BUS "B" latch when this register is written.

3.2.8 Bus Control/Slave Status Register (07)

This eight-bit read/write register is used to receive the Bus Control octet during a Bus Control sequence and receives the Slave Status octet from the drive processor during an Ending Status sequence. Reading this register resets the Bus Control Available Bit in the Interrupts Register. Writing to this register sets a flag, causing the IPC State Sequencer to exit the IPI Information Transfer Sequence and begin an Ending Status Sequence.

3.3 Serdes Bus Interface

The Serdes Bus Interface is the data path over which Data Bus Controls, Ending Status from Data Bus Controls, and disk read/write data are passed during the execution of drive read/write data controls.

3.4 IPI Bus Buffers

The IPI Bus Buffer logic routes the information contained on Bus "A" or Bus "B", during the proper bus states, to/from the proper internal IPC logic component under the direction of the State Sequencer.

3.5 State Sequencer

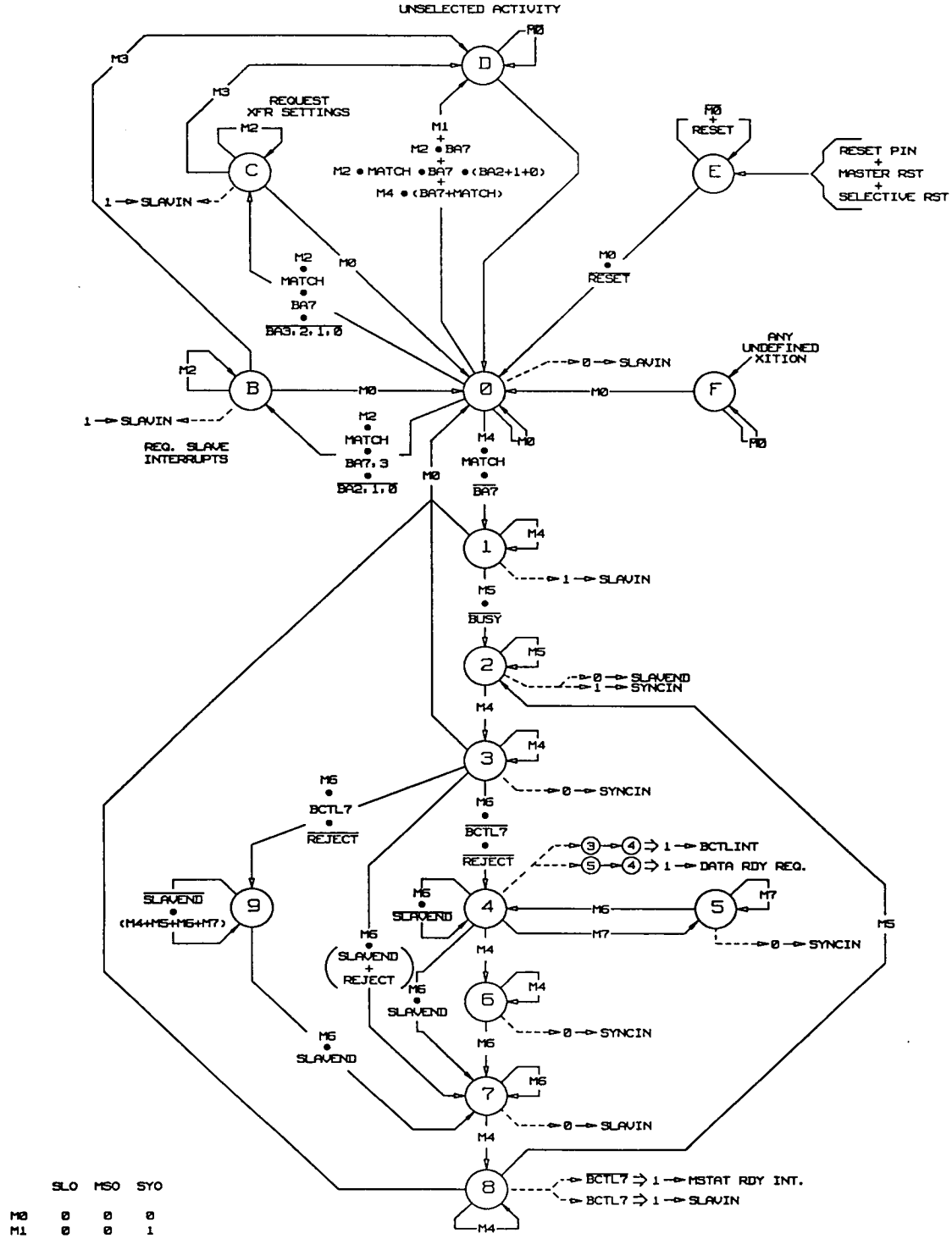
The IPC State Sequencer interprets the IPI Bus State transitions and directs the flow of data to/from the IPI Bus Buffers to the proper IPC logic component, and/or performs the required action dictated by a particular State Transition Sequence. Bus Controls, when received, are examined to determine their type. Data Bus Controls are routed to the Serdes Bus Interface for execution by the Serdes/Formatter. Command/Response Bus Controls are routed to the Bus Control register in the register array and the Bus Control available bit set in the Interrupt Register.

The action taken by the State Sequencer for any transition of the IPI control signals is defined in the IPC State Diagram (Figure 3).

3.6 Disk Data FIFO

The Disk Data FIFO buffers write data until the Serdes/Formatter is ready to begin accepting it. The FIFO can hold up to eight bytes of disk write data. The Serdes/Formatter Circuit begins requesting write data before it is required to allow for cable and controller delays. The FIFO provides a place to store this data should it arrive from a "close" controller sooner than required.

The FIFO is a fall-through design that is organized as four 16-bit registers. Write data is loaded into the FIFO from the IPI Bus on the active going edge of the Sync Out signal. Data is removed from the FIFO one byte at a time under the control of the Write Data Strobe and Sync signals of the Serdes Bus. The first byte from the FIFO is removed from the BUS A side on the falling edge of the Write Data Strobe signal. On the next Write Data Strobe pulse, a byte is removed from the BUS B side, allowing a 16-bit data value to fall through. Subsequent bytes are removed from alternate sides of the FIFO, with every access of the Bus B side allowing the FIFO to advance.



	SLO	MSO	SYO
M0	0	0	0
M1	0	0	1
M2	0	1	0
M3	0	1	1
M4	1	0	0
M5	1	0	1
M6	1	1	0
M7	1	1	1

Figure 3
Interface Protocol Circuit
State Diagram

4.0 OPERATIONAL DESCRIPTION

The Interface Protocol Circuit is responsible for interpreting all IPI Bus state transitions and state sequences. The drive processor is involved only when a Command/Response Bus Control, Command/Response Parameter, Master Status, or Slave Status Octet is received or required, and/or if an error occurs.

The sections that follow, provide a brief description of action taken by the Interface Protocol Circuit in the interpretation of each IPI Bus State Sequence. References to states and octets are references to IPI Bus states and octets. The reader is assumed to be familiar with contents of the ANSI IPI Physical Layer document.

4.1 Request Interrupts Sequence

This sequence is handled entirely by the Interface Protocol Circuit without intervention from the drive processor. The sequence starts from the IDLE state, and proceeds to the REQUEST state after the Controller places a Request Interrupts octet on Bus A and asserts Master Out. Each drive IPC decodes the Bus A octet and compares the contents, bits 0-6, with the contents of its Request Interrupts Register. If a match occurs in any bit position, the drive IPC responds with its Bit Significant Response: it asserts the Bus B line associated with the address contained in the Address Register. The Bit Significant Response is dynamic (can change as the conditions contained in the Request Interrupts Register change) for as long as the Controller remains in the REQUEST state. The sequence ends when the controller negates the Master Out signal and returns to the IDLE state. See Figure 6 for timing.

4.2 Request Facility Interrupts Sequence

Facility interrupts are not supported by IPI level 2 and the IPC does not respond to this sequence.

4.3 Request Slave Interrupts Sequence

This sequence is handled entirely by the Interface Protocol Circuit without intervention from the drive processor. The sequence starts from the IDLE state, and proceeds to the REQUEST state after the Controller places a Request Slave Interrupts octet on Bus A and asserts Master Out. Each drive IPC decodes the Bus A octet, and the addressed IPC advances the bus to the REQUACK state by placing the contents of its Request Interrupts Register on Bus B and asserting the Slave In signal. The contents of Bus B can change dynamically as the conditions in the Request Interrupts Register change, as long as the Controller remains in the REQUACK state. The Controller initiates termination of the sequence by negating Master Out and advancing to the DESEL state. The addressed IPC ends the sequence by negating Slave In and returning to the IDLE state. See Figure 7 for timing.

4.4 Request Transfer Settings Sequence

This sequence is handled entirely by the Interface Protocol Circuit without intervention from the drive processor. The sequence starts from the IDLE state, and proceeds to the REQUEST state after the Controller places a Transfer Settings octet on Bus A and asserts Master Out. Each drive IPC decodes the Bus A octet, and the addressed IPC advances the bus to the REQUACK state by placing the IPI level-2 Transfer Settings octet on Bus B and asserting the Slave In signal. The Controller initiates termination of the sequence by negating Master Out and advancing to the DESEL state. The addressed IPC ends the sequence by negating Slave In and returning to the IDLE state. See Figure 7 for timing.

4.5 Master Reset Sequence

This sequence is handled entirely by the Interface Protocol Circuit without intervention from the drive processor. The sequence starts from any state and begins when the Controller negates Select Out, if active. The addressed IPC, if driving the BUS A lines, is expected to release them. After a delay, the Controller releases Master Out and Sync Out and places a Master Reset octet on BUS A. After an additional delay, the Controller negates Master Out, asserts Sync Out, and enters the MAINT state. Each drive IPC decodes the Bus A octet and disables its Slave In and Sync In drivers. The sequence ends when the controller negates the Master Out signal and returns to the IDLE state. Upon entering the IDLE state, each drive performs the action required by the received Master Reset octet.

4.6 Selective Reset Sequence

This sequence is handled entirely by the Interface Protocol Circuit without intervention from the drive processor. The sequence starts from the IDLE state, and proceeds to the REQUEST state after the Controller places a Selective Reset Control octet on Bus A. Each drive IPC decodes the Bus A octet, and the addressed IPC, if enabled and functional, advances the bus to the REQUACK state by placing the appropriate response octet on Bus B and asserting the Slave In signal. Up to this point, a functional IPC will interpret the octet on Bus A as either a Request Slave Interrupts, a Request Facility Interrupts, or a Request Transfer Settings octet. After a delay, regardless of whether the addressed drive IPC responds with Slave In, the controller advances to the RESETSEL2 state by asserting Sync Out. The addressed IPC, if enabled and functional, advances to the RESETSEL1 state by releasing the BUS B lines and the Slave In signal. At this point the addressed drive IPC, enabled and functional or not, recognizes the octet on BUS A as a Selective Reset Control octet and performs the required action. After a suitable delay to allow the addressed drive to reset, the Controller advances to the REQUEST state by negating Sync Out. The addressed IPC responds by asserting the Slave In signal and advancing to the REQUACK state. The Controller initiates termination of the sequence by negating Master Out and advancing to the DESEL state. The sequence ends when the addressed IPC returns to the IDLE state by negating Slave In.

4.7 Selection Sequence

This sequence is handled entirely by the Interface Protocol Circuit without intervention from the drive processor. The sequence starts from the IDLE state, and proceeds to the SELECT state after the Controller places a Selection octet on Bus A and asserts Select Out. Each drive IPC decodes the Bus A octet. The sequence ends when the addressed IPC advances the bus to the SLAVACK state by asserting Slave In and, if not busy, placing its Bit Significant Response on BUS B. The addressed IPC will not respond if the Selection octet has bits 1, 2, or 3 set, or if the octet on BUS A is a Facility Selection octet. If the addressed IPC does not respond, the Controller must end the sequence by negating Select Out and returning to the IDLE state. See Figure 4 for timing.

4.8 De-selection Sequence

This sequence is handled entirely by the Interface Protocol Circuit without intervention from the drive processor. The sequence starts from the SLAVACK state, and proceeds to the DESEL state after negating Select Out. The sequence ends when the selected drive IPC negates Slave In and returns to the IDLE state. See Figure 5 for timing.

4.9 Bus Control Sequence

This sequence is handled entirely by the Interface Protocol Circuit without intervention from the drive processor. The sequence starts from the SLAVACK state and proceeds to the BUSCTL state after the Controller places a Bus Control octet on Bus A and asserts Sync Out. The selected drive IPC advances to the BUSACK state by latching the contents of BUS A into the Bus Control Register, placing the Bus Acknowledge octet (all zeros) on BUS B, and asserting Sync In. The Controller initiates termination of

the sequence by negating Sync Out and advancing to the MASTEND state. The addressed IPC ends the sequence by negating Sync In and returning to the SLAVACK state. See Figure 8 for timing.

4.10 Interlocked Input Sequence

This sequence starts from the SLAVACK state, and must have been immediately preceded by a Bus Control Sequence for the initial state transition to be considered valid. The sequence starts when the Controller asserts Master Out, advancing to the XFRRDY state. The selected drive IPC has previously received a Response Bus Control (bit 7 of the Bus Control octet was zero, and bit 6 was set). The selected drive IPC responds by setting the Bus Control Available bit in the Interrupt Register. No further action will take place until the drive processor responds to this interrupt. The drive processor is expected to read the contents of the Bus Control Register, decode the Response Control, and reset the Bus Control Available Interrupt bit. The processor has two choices at this point: output a Response parameter or output a Slave Status octet.

To output a Response parameter, the processor writes the high-order byte of the parameter in the Bus A register and the low-order parameter in the Bus B register. When a write occurs to the Bus B register, the IPC sequencer clears the XFRRDY state and advances to the XFRST state by asserting Sync In. The Controller accepts the Response parameter and either terminates or continues the information transfer. To continue the transfer it asserts Sync Out, advancing to the XFRRES state. The selected IPC respods by negating Sync In and enters the XFREND state. The Controller negates Sync Out arriving at XFRRDY. The selected IPC sets the Data Ready/Request status bit and waits for the processor to respond.

The processor has the same two choices it had at the beginning of the transfer loop: output a Response parameter or output a Slave Status octet.

If the last parameter transferred was the last parameter required by the Response Control, the processor would terminate the information transfer at this point by writing a Slave Status octet to the Slave Status Register. When a write occurs to the Slave Status Register, the IPC sequencer exits the XFRRDY state and advances to the SLAVEND state. The Controller responds by placing a Master Status octet on BUS A, negating Master Out, and entering the SELECT state. The selected IPC latches the contents of BUS A into the Master Status Register, sets the Master Status Available status bit, and waits for the processor to accept the octet. When the processor reads the Master Status Register, the IPC ends the sequence by outputting the contents of the Slave Status Register to BUS B, asserting Slave In, and entering the SLAVACK state.

The controller could terminate the information transfer while in the transfer loop, in which case the ending sequence would be somewhat different. This option exists when the bus is in the XFRST state of the Interlocked Input sequence. After accepting the Response parameter, the controller negates Master Out, beginning a master end sequence. The selected IPC negates Sync In and enters the SLAVACK state. The Controller asserts Master Out and enters the XFRRDY state, after which the selected IPC responds by negating Slave In and entering the SLAVEND state. The Controller then places a Master Status octet on BUS A, negates Master Out, and enters the SELECT state. The select IPC sets the Master Status Available status bit and waits for the processor to accept the octet. When the processor reads the Master Status Register, the status bit is reset. The processor writes a Slave Status octet to the Slave Status Register, allowing the IPC to end the sequence. The IPC outputs the contents of the Slave Status Register to BUS B, asserts Slave In, and enters the SLAVACK state.

4.11 Interlocked Output Sequence

The Interlocked Output Sequence is very similar to the Input Sequence. However, for the sake of clarity a detailed description of the sequence is included below.

This sequence starts from the SLAVACK state and must have been immediately preceded by a Bus Control Sequence for the initial state transition to be considered valid. The sequence starts when the Controller asserts Master Out, advancing to the XFRRDY state. The selected drive IPC has previously received a Command Bus Control (bits 7 and 6 of the Bus Control octet were zero). The selected drive IPC responds by setting the Bus Control Available bit in the Interrupt Register. No further action will take place until the drive processor responds to this interrupt. The drive processor is expected to read the contents of the Bus Control Register, decode the Command Control, and reset the Bus Control Available Interrupt bit. The processor has two choices at this point: read a Command parameter or output a Slave Status octet.

If the Parity Error status bit was set at the time the Bus Control Available Interrupt occurred, the processor would output a Slave Status octet at this time as described below.

To input a Command parameter, the processor performs a dummy write to the Bus B register. When a write occurs to the Bus B register, the IPC sequencer clears the XFRRDY state and advances to the XFRST state by asserting Sync In. The Controller can either terminate or continue the information transfer. To continue to transfer, it places the Command parameter on BUS A and B, asserts Sync Out, and advances to the XFRRES state. The IPC latches the parameter in the Bus A and Bus B registers, negates Sync In, and enters the XFREND state. The Controller negates Sync Out, arriving at XFRRDY. The IPC sets the Data Ready/Request status bit and waits for the processor to respond.

The processor has the same two choices it had at the beginning of the transfer loop: input a Command parameter or output a Slave Status octet.

If the last parameter transferred was the last parameter required by the Command Control, the processor would terminate the information transfer at this point by writing a Slave Status octet to the Slave Status Register. The sequence from this point is identical with the corresponding part of the Interlocked Input Sequence and is described in detail under that section.

The controller could terminate the information transfer while in the transfer loop, in which case the ending sequence would be somewhat different. This option exists when the bus is in the XFRST state of the Interlocked Input sequence. After recognition of the XFRST state, the controller negates Master Out, beginning a master end sequence. The sequence from this point is identical with the corresponding part of the Interlocked Input Sequence and is described in detail under that section.

4.12 Non-Interlocked Input Sequence

This sequence is handled entirely by the Interface Protocol Circuit, in conjunction with the Serdes/Formatter Circuit, without intervention from the drive processor. This sequence starts from the SLAVACK state and must have been immediately preceded by a Bus Control Sequence for the initial state transition to be considered valid. The sequence starts when the Controller asserts Master Out, advancing to the XFRRDY start, and the selected drive IPC has previously received a read type Data Bus Control (bit 7 of the Bus Control octet was set, and bit 6 was cleared). The drive IPC responds by setting Arbitrate Out and testing the state of Arbitrate In. If Arbitrate In is inactive, the IPC enables the contents of the Bus Control Register to the Serdes Data Bus and asserts the Bus Control Strobe signal. The Serdes/Formatter Circuit latches and decodes the Bus Control. Upon recognizing it as a read data operation, the SFC asserts the Ending Status Strobe signal to gain control of the Serdes data bus. The IPC responds by releasing the data bus and negating Bus Control Strobe.

When disk read data becomes available, the SFC Sync signal will go active and the data byte present on the Serdes bus will be latched into the BUS A Holding register. When Sync goes inactive, the BUS A and BUS B registers will go transparent, allowing the contents of the Holding register to pass through the BUS A register onto BUS A and allowing the contents of the Serdes bus to pass through the BUS B register onto BUS B. On the next Sync pulse, the BUS A and BUS B registers latch the data present at their inputs.

The SFC Sync signal is divided by two to generate the IPC Sync In signal during the non-interlocked disk read data transfer. This signal is NANDed with the SFC Sync signal to generate the latch enable for the BUS A and BUS B registers. The latch enable signal for the Bus "A" Holding register is Sync divided by two, but leading Sync In by 90 degrees.

When the disk read operation is over, the SFC pads the transfer by one byte, if necessary, to ensure that an even number of Sync pulses were generated. The SFC then places a Slave Status byte on the Serdes bus and negates Ending Status Strobe. The IPC latches the value of the Serdes bus into the Slave Status Register, and enables its Serdes bus drivers in preparation for the next Data Bus Control. A complete Slave Status octet is formed by setting 6 and resetting bit 7 of the Slave Status register, if a parity error occurred during the transfer.

At this point, the IPC negates Slave In and enters the SLAVEND state. The Controller places a Master Status octet on BUS A and negates Master Out, entering the SELECT state. The IPC ignores the contents of BUS A, places the contents of the Slave Status Register on BUS B, asserts Slave In, and enters the SLAVACK state, ending the sequence. See Figure 8 for timing.

4.14 Non-Interlocked Output Sequence

This sequence is handled entirely by the Interface Protocol Circuit, in conjunction with the Serdes/Formatter Circuit, without intervention from the drive processor. This sequence starts from the SLAVACK state and must have been immediately preceded by a Bus Control Sequence for the initial state transition to be considered valid. The sequence starts when the Controller asserts Master Out, advancing to the XFRRDY state, and the selected drive IPC has previously received a write/verify type Data Bus Control (bits 6 and 7 of the Bus Control octet were set). If the Arbitration Mode Bit (Status Register bit-3) is set, the IPC responds by setting Arbitrate Out and testing the state of Arbitrate In. If Arbitrate In is inactive, the IPC enables the contents of the Bus Control Register to the Serdes Data Bus and asserts the Bus Control Strobe signal. The Serdes/Formatter Circuit latches and decodes the Bus Control. Upon recognizing it as a write/verify disk operation, the SFC locates the beginning of the required sector and begins transmitting Sync pulses at the disk data rate. After six byte times the SFC pauses transmission of Sync pulses until Sync Byte has been written or located, after which it begins transmission of Sync pulses again until the transfer is complete.
The IPC divides the SFC Sync signal by two to generate the Sync In signal during the non-interlocked disk write/verify data transfer. When a Sync Out pulse arrives from the controller, the contents of BUS A and BUS B are latched into the top of the 16- bit by four word disk write/verify data FIFO.

After the falling edge of the seventh Sync pulse, the SFC asserts the Write Data Strobe signal, enabling data from the bottom of the write/verify data FIFO onto the Serdes data bus. The SFC latches the data on the next rising edge of Sync. The data transfer alternates between the BUS A and BUS B sides of the FIFO on the rising edge of Sync, starting with the BUS A side. When the SFC is seven bytes from the end of the write/verify operation, it negates Write Data Strobe. The IPC reponds by terminating the Sync In pulse train on the next falling edge of Sync In.

When the end of the write/verify operation occurs, the SFC asserts Ending Status Strobe to acquire the Serdes data bus. The IPC responds by releasing the data bus and negating Bus Control Strobe. The SFC then places a Slave Status byte on the Serdes bus, and negates Ending Status Strobe. The IPC latches the value of the Serdes bus into the Slave Status Register, and enables its Serdes bus drivers in preparation for the next Data Bus Control. A complete Slave Status octet is formed by setting bit 6 and

resetting bit 7 of the Slave Status register, if a parity error occurred during the transfer.

At this point, the IPC negates Slave In and enters the SLAVEND state. The Controller places a Master Status octet on BUS A and negates Master Out, entering the SELECT state. The IPC ignores the contents of BUS A, places the contents of the Slave Status Register on BUS B, asserts Slave In, and enters the SLAVACK state, ending the sequence. See Figure 8 for timing.

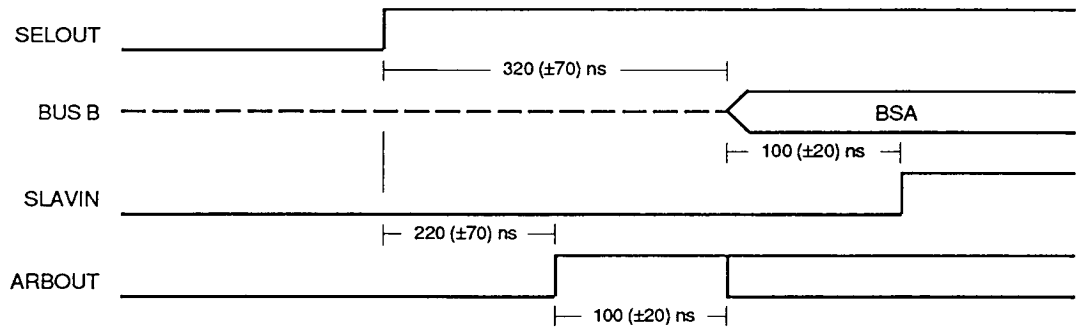


Figure 4
Selection Sequence

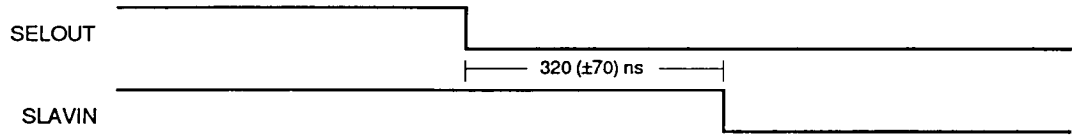


Figure 5
Deselection Sequence

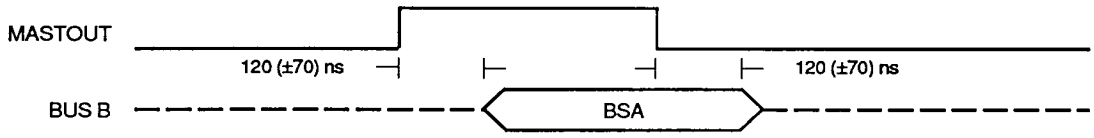


Figure 6
Request InterruptsSequence

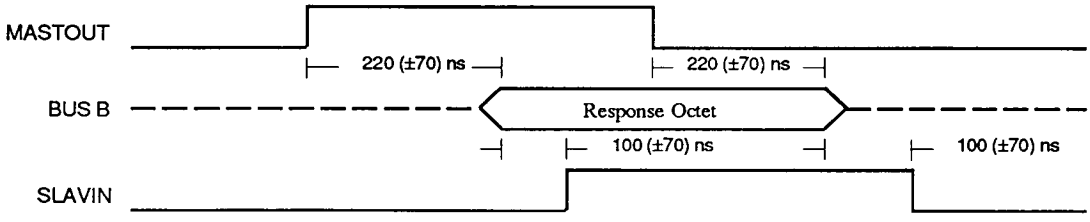


Figure 7
Interlocked Request Sequences

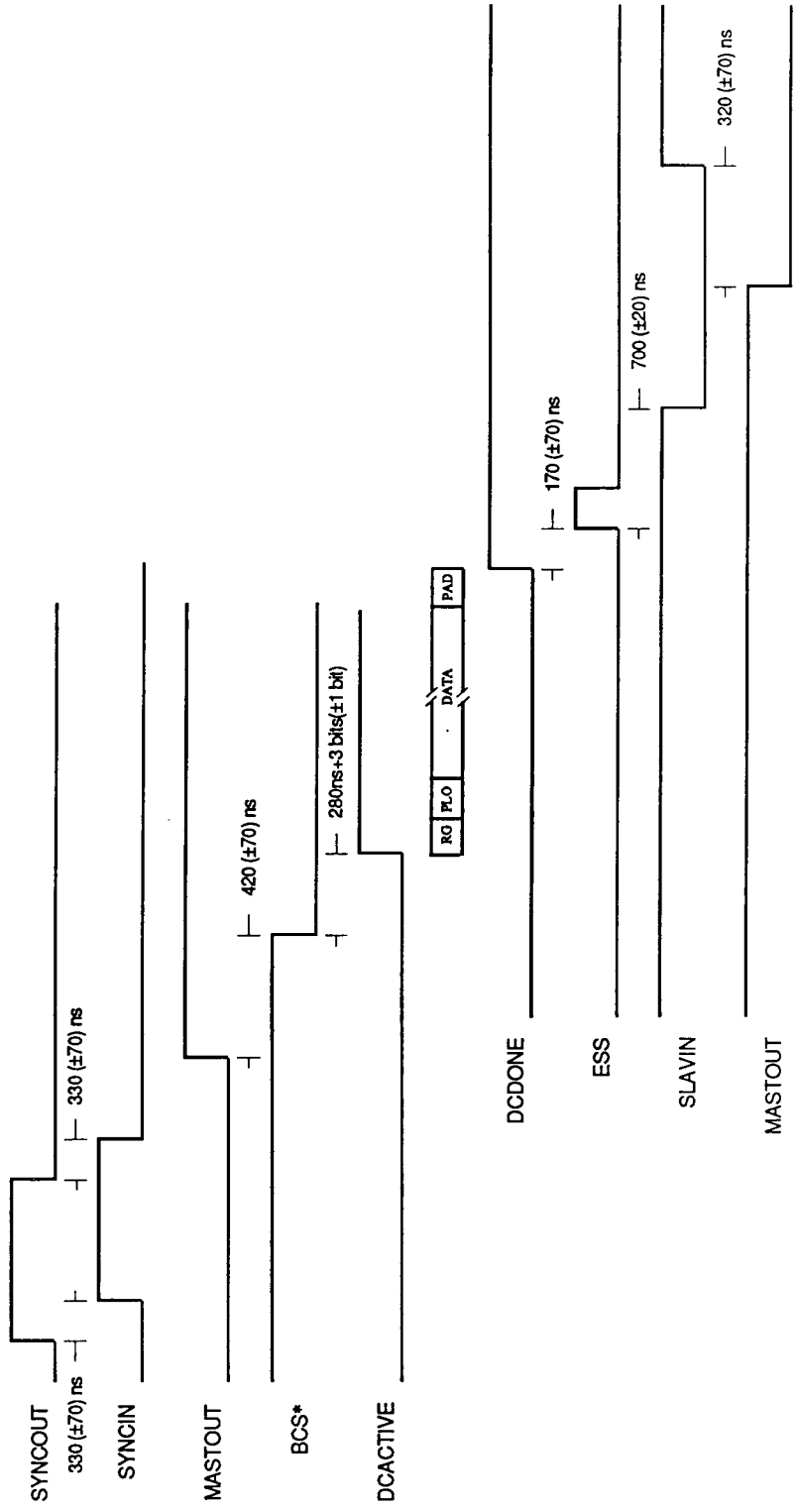


FIGURE 8
DATA CONTROL
SETUP AND TERMINATION

5.0 BUS OCTETS

This section defines the format of the Bus Octets received from the Controller on BUS A and the Bus Octets transmitted by the Drive on BUS B.

5.1 Request Interrupts Octet

The Request Interrupts octet is received in the REQUEST state and is addressed to all drives. The circuit responds with its bit significant address, if any of the conditions that have a 1-bit set in the octet match the corresponding bit of the Request Interrupts Register.

Bit	Description
7	0
6	Busy Status
5	Ready Status
4	Power Fail Alert
3	Power On Status
2	Class 3 Interrupt
1	Class 2 Interrupt
0	Class 1 Interrupt

5.2 Request Facility Interrupts Octet

This octet is received in the REQUEST state and is used to request facility interrupts. Facility interrupts are not supported by IPI level-2, and the IPC will not respond to a Facility Interrupts Octet while in the REQUEST state.

Bit	Description
7	1
6-4	Slave Select Code
3	Facility Range
2-0	Interrupt Class (at least one bit must be set)

5.3 Request Slave Interrupts Octet

This octet is received in the REQUEST state and is used to request slave interrupts. If the Drive Select Code matches the contents of the Port Address register, the circuit places a Slave Interrupts Octet on BUS B and asserts SLAVE IN.

Bit	Description
7	1
6-4	Drive Select Code
3	1
2-0	0

5.4 Request Transfer Settings Octet

This octet is received when in the REQUEST state and is used to request transfer settings. If the Drive Select Code matches the contents of the Port Address register, the circuit places its Transfer settings Octet on BUS B and asserts SLAVE IN.

Bit	Description
7	1
6-4	Drive Select Code
3	0
2-0	0

5.5 Selective Reset Octet

The Selective Reset octet is interpreted in the RESETSEL1 and RESETSEL2 states. If the Drive Select Code matches the contents of the Port Address register, circuit actions depend on the setting of bits 2-0.

Bit	Description
7	1
6-4	Drive Select code
3	Release Slave Drivers
2	Power-on Reset
1	Interrupt microprocessor (Logical Reset)
0	Reset IPC (Physical Reset)

5.6 Selection Octet

The Selection octet is received in the SELECT state and it addresses a drive. If the Drive Select Code in the octet matches the contents of the Port Address register, the circuit responds by asserting SLAVE IN. If the port is not busy, the circuit reponds by asserting its radial address bit on BUS B. An override of any reserve condition condition takes place when bit 0 of the octet is set, causing the drive to become selected.

Bit	Description
7	0
6-4	Drive Select Code
3	0
2	0
1	0
0	Priority Select

5.7 Command/Response Bus Control Octet

The Command/Response octet is received in the BUSCTL state and is saved in the Bus Control register until the XFRRDY state is entered. At that time a processor interrupt is generated.

Bit	Description
7	0
6	1=Information In (Response), 0= Information Out (Cmd)
5	0
4	0
3	0
2-0	Type Code

5.8 Data Bus Control Octet

The Data Control octet is received in the BUSCTL state, and is saved in the Bus Control register until the XFRRDY state is entered. At that time it is transferred to the Serdes-Formatter for execution.

Bit	Description
7	1
6	1=Information In (Read), 0=Information Out (Write)
5	0 reserved
4	Head Advance Control
3-0	Type Code

5.9 Master Status Octet

The Master Status is received in the SELECT state after an information transfer. If the information transfer was a command or response, the status octet is read by the processor from the Master Status register.

Bit	Description
7	Successful Information Transfer
6	Bus Parity Error
5-0	0

5.10 Slave Interrupts Octet

This octet is output on BUS B, during the REQUEST state, if the Drive Select Code of the Request Slave Interrupts Octet matches the contents of the Port Address register.

Bit	Description
7	0
6	Busy Status
5	Ready Status
4	0
3	Priority Select
2	Class 3 Interrupt
1	Class 2 Interrupt
0	Class 1 Interrupt

5.11 Slave Ending Status Octet

The Slave Status octet is output during SLAVACK state after an information transfer. If the information transfer was a command or response, the status octet is loaded into the Slave Status register by the processor. If the information was disk read/write data, the status octet is developed by the Interface Protocol Circuit and Serdes/Formatter Circuit.

Bit	Description
7	Successful Information Transfer
6	Bus Parity Error
5	Odd Byte Transfer
4	Time dependent operation (commands only)
3-0	Ending Status

6.0 DUAL PORT

The Interface Protocol Circuit may be used in the implementation of dual port disk drives. The Interface Protocol Circuits are connected to a single Serdes/Formatter Circuit and a drive processor. The drive processor may be one devoted exclusively to the interface or an already existing processor (e.g. seek microprocessor). One or two processors devoted to the interface could be used to service the two Interface Protocol Circuits depending, on the level of performance desired.

The Interface Protocol Circuits are capable of arbitrating between each other on behalf of the drive ports they each serve. When both ports require the use of the drive simultaneously, they must arbitrate to determine which port will have access and which port will receive a drive busy indication. IPI provides for two means of indicating a busy condition: busy condition may be reported during selection, or after selection at the time of receipt of a Bus Control. Thus there are two methods of supporting dual port access to a disk drive.

One method would allow only a single port to gain selection of the drive at any given time. A port attempting to select, when the alternate port already has selected or reserved the drive, would receive a busy indication at the time of selection. This method of dual port operation is sometimes referred to as the "Physical Level Busy" mode, since it relies on the physical interface selection busy mechanism.

The second method allows both ports to gain selection, but will allow only one port to gain access to the drive HDA at any given time. A port attempting to execute a drive read/write Data Bus Control, or any Command/Response Bus Control requiring access to the drive HDA, when the alternate port already has reserved or gained access to the drive HDA, will receive a drive busy indication through the Slave Status Octet. This method of dual port operation is sometimes referred to as the "Logical Level Busy" mode, because the drive does not appear physically busy at the time of selection, but rather appears logically busy, depending on the particular Bus Control issued. Logical Level Dual Port allows a certain degree of concurrent Bus Control execution, depending on the implementation, single or dual drive processors, and the type of Bus Control issued by each port.

6.1 Physical Level Dual Port

In this mode of operation, the drive indicates during the selection sequence that it is busy with an alternate port operation or is reserved to the alternate port. To accomplish this mode of operation, the drive processor:

clears the Arbitrate Mode bit in the Configuration Register;
clears the Force Selection Busy bit in the Control Register;
clears the Command/Response Busy bit in the Control Register;
clears the Data Control Busy bit in the Control Register;
sets the Enable Port bit in the Control Register

of each port Interface Protocol Circuit during power-up initialization.

After it has been enabled, the IPC associated with a particular port will not respond to any IPI state changes until it observes the IPI Bus in the Idle state.

The clearing of the Arbitrate Mode bit assures that only one port gains selection of the drive at any given time.

If a port gains selection and issues a Disable Alternate Port Function Code, the drive processor clears the Enable Port Bit in the alternate port IPC until an Enable Alternate Port Function Code is executed by the same port.

If a port gains selection and issues a Reserve Command, the drive processor sets the Force Selection

Busy bit in the alternate port IPC until the reserving port issues a Release Command. This allows the reserving port to deselect, while a selection busy condition is maintained on the alternate port.

If a port gains selection and issues a Command requiring the Time Dependent Operation (TDO) bit to be set in the Slave Status octet, the drive processor sets the Force Selection Busy Bit in the alternate port IPC until the Command is complete. This prevents the alternate port from selecting the drive, should the selected port deselect before the command has completed execution. The drive processor must also set the Force Command/Response Busy and Force Data Control Busy bits in the selected port IPC prior to releasing it from the XFRRDY state with a Slave Status octet, to prevent that port from issuing any further Bus Controls until the current command completes execution. Upon command completion, the drive processor posts a Command Complete Interrupt (IPI Class 1 interrupt), clears the Force Data Control Busy and Force Command/Response Busy bits in the issuing port IPC, and clears the Force Port Busy Bit in the alternate port IPC.

6.2 Logical Level Dual Port

In this mode of operation, the drive indicates during the ending status sequence that it is busy with an alternate port operation or is reserved to the alternate port. To accomplish this mode of operation, the drive processor:

sets the Arbitrate Mode bit in the Status Register;
clears the Force Selection Busy bit in the Control Register;
clears the Command/Response Busy bit in the Control Register;
clears the Data Control Busy bit in the Control Register;
sets the Enable Port bit in the Control Register

of each port Interface Protocol Circuit during power-up initialization.

After it has been enabled, the IPC associated with a particular port will not respond to any IPI state changes until it observes the IPI Bus in the Idle state.

The setting of the Arbitrate Mode bits allows both ports to gain selection of the drive at the same time, but assures that only one port will gain access to the Serdes/Formatter Circuit via a Data Bus Control at any given moment. It remains for the drive processor to prevent simultaneous accessing of the drive HDA by the ports through Command/Response Bus Controls.

When a port successfully issues a Disable Alternate Port Function Code, the drive processor clears the Enable Port bit and sets the Force Data Control Busy and Force Command/Response Busy bits in the alternate port IPC until an Enable Alternate Port Command is executed by the same port. This action prevents the alternate port from issuing any further Bus Controls if it should be selected at the time the Alternate Port command was received. When the alternate port de-selects, the port becomes disabled.

If a port successfully issues a Reserve command, the drive processor sets the Force Data Control Busy and Force Command/Response Busy bits in the alternate port IPC until a Release command is executed by the same port. This allows the reserving port to deselect, while a "logical busy" condition is maintained on the alternate port.

If a port successfully issues a Command Bus Control requiring the Time Dependent Operation (TDO) bit to be set in the Slave Status octet, it becomes the active port. The drive processor sets the Force Command/Response Busy bit in the alternate port IPC to force a drive busy Slave Status octet in response to any Command/Response Bus Controls from that port.

The drive must also set the Force Command/Response Busy and Force Data Control Busy bits in the active port IPC, to force a drive busy Slave Status octet in response to any additional Command/Response Controls, before releasing it from the XFRRDY state.

Upon command completion, the drive processor posts a Command Complete Interrupt (IPI Class 1 interrupt) to the active port, and clears the Force Data Control Busy and Force Command/Response Busy bits (if set) in the active port IPC (if set) and the alternate port IPC (if set).