



*Personal Computer  
Hardware Reference  
Library*

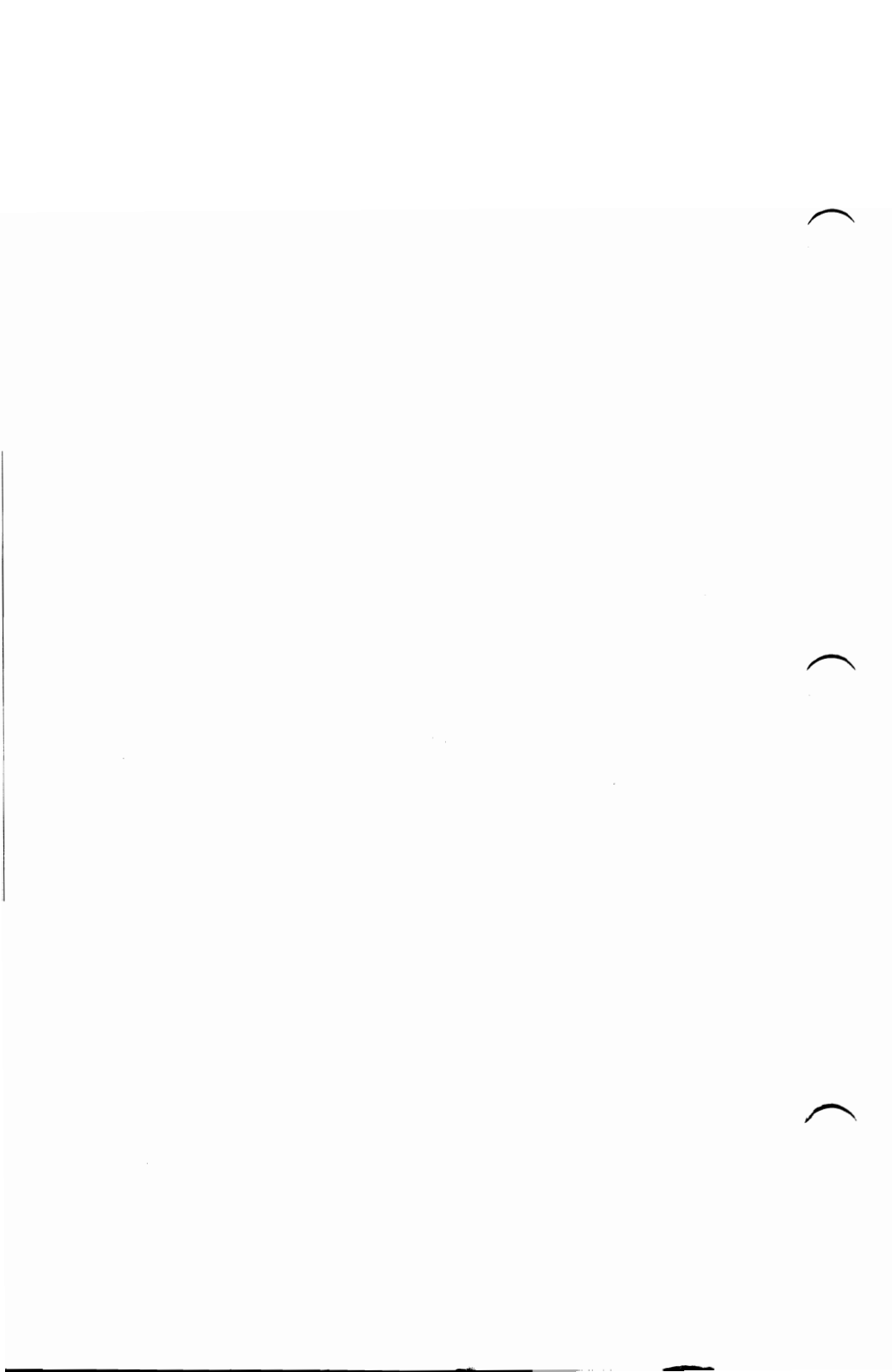
---

**IBM Personal Computer  
General Purpose  
Interface Bus Technical  
Reference**

6138155

August 15, 1984

© Copyright IBM Corporation 1984



# Contents

Description .....	1
Major Components .....	4
Data-Bus Buffer .....	5
Control-Signal Buffer .....	5
DMA Circuitry .....	6
Interrupt Circuitry .....	7
Address Decode Logic .....	12
$\mu$ PD7210 Talker/Listener/Controller (TLC) ..	13
GPIB Transceivers .....	30
Interface Control Logic .....	31
Programming Considerations .....	33
The GPIB Adapter as a Controller .....	35
The GPIB Adapter as a Talker or Listener .....	39
Programmed Implementation .....	39
Addressed Implementation .....	39
Sending and Receiving Commands .....	43
Programmed I/O Data Transfer .....	43
DMA Data Transfer .....	43
Sending END or EOS .....	44
Stopping on an END or EOS .....	44
Serial Polls .....	45
Conducting Serial Polls .....	45
Responding to a Serial Poll .....	46
Parallel Polls .....	47
Configuring for a Parallel Poll .....	48
Conducting a Parallel Poll .....	50
Disabling Parallel Polling .....	50
Responding to a Parallel Poll .....	51
Interrupts .....	52
Programming the Interrupt Controller .....	54
DMA Transfers .....	55
Programming the DMA Controller .....	56
Interface .....	57
Connector Specifications .....	58
Jumper Positions .....	59
Adapter Selection .....	60

DMA Channel Selection .....	61
Interrupt-Request Selection .....	62
Interrupt-Acknowledge Selection .....	63
Specifications .....	65
Logic Diagram .....	67

<b>Index .....</b>	<b>Index-1</b>
--------------------	----------------

# Description

The IBM General Purpose Interface Bus (GPIB) Adapter connects IBM Personal Computer products (PC) to a general-purpose interface bus (bus) and is designed according to the specifications of the following industry standards as understood and interpreted by IBM as of September 1983: ANSI/IEEE Std. 488-1978 (includes supplement IEEE Std. 488A-1980).

The GPIB Adapter makes it possible to use the PC as a controller for a GPIB test and measurement system. The GPIB Adapter implements bus interface functions for communication with GPIB devices, and implements device functions for communication with the PC central processor and memory.

The following lists interface function codes supported by the IBM GPIB Adapter.

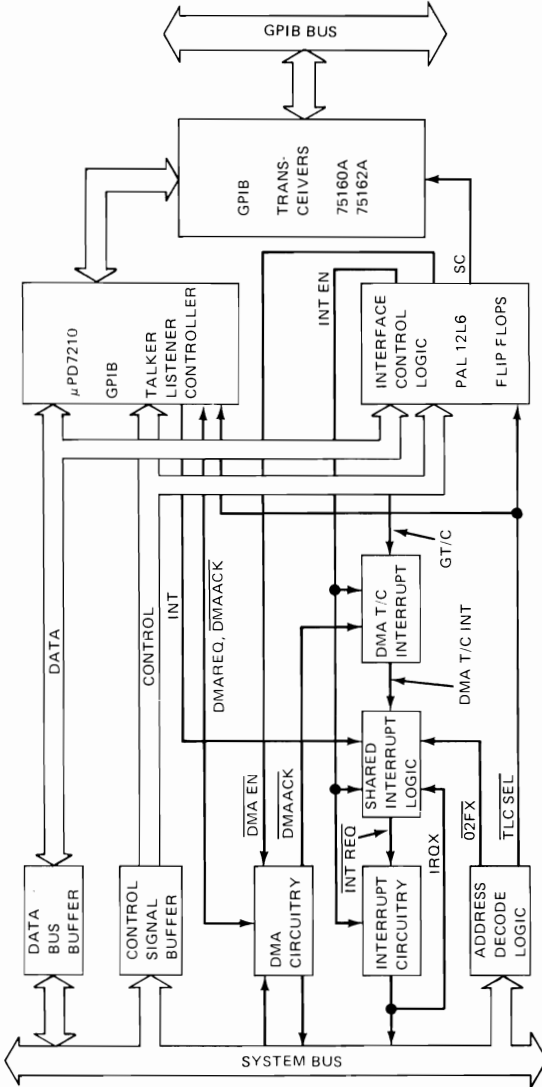
<b>Code</b>	<b>Description</b>
<b>SH1</b>	Complete source-handshake capability
<b>AH1</b>	Complete acceptor-handshake capability:  data-accepted and ready-for-data (RFD) holdoff on certain events
<b>T5</b>	Complete talker capability:  Basic talker Serial poll Talk-only mode Unaddressed on my-listen-address (MLA) Send end message (END) Send end-of-string message (EOS) Dual primary addressing

<b>TE5</b>	<p>Complete extended-talker capability:</p> <ul style="list-style-type: none"> <li>Basic extended talker</li> <li>Serial poll</li> <li>Talk-only mode</li> <li>Unaddressed on my-secondary-address (MSA) or listener-primary-addressed-state (LPAS)</li> <li>Send END or EOS</li> <li>Dual extended addressing with software assist</li> </ul>	)
<b>L3</b>	<p>Complete listener capability:</p> <ul style="list-style-type: none"> <li>Basic listener</li> <li>Listen-only mode</li> <li>Unaddressed on my-talk-address (MTA)</li> <li>Detect END or EOS</li> <li>Dual primary addressing</li> </ul>	
<b>LE3</b>	<p>Complete extended-listener capability:</p> <ul style="list-style-type: none"> <li>Basic listener</li> <li>Listen-only mode</li> <li>Unaddressed on MSA or talker-primary-addressed-state (TPAS)</li> <li>Detect END or EOS</li> <li>Dual extended addressing with software assist</li> </ul>	)
<b>SR1</b>	Complete service request capability	
<b>RL1</b>	Complete remote and local capability with software interpretation	
<b>PP1</b>	Remote parallel poll configuration	
<b>PP2</b>	Local parallel poll configuration with software assist	)

- DC1** Complete device-clear capability with software interpretation
- DT1** Complete device-trigger capability with software interpretation
- C1-5** Complete controller capability:
  - System controller
  - Send interface-clear (IFC) and take control
  - Send remote-enable (REN)
  - Respond to service-request (SRQ)
  - Send interface messages
  - Receive control
  - Pass control
  - Parallel poll
  - Take control synchronously or asynchronously
- E1/2** Three-state bus drivers with automatic switch to open-collector drivers during parallel poll

# Major Components

The following block diagram shows the major components of the GPIB Adapter.





## Data-Bus Buffer

The data-bus buffer transfers the data between the system's input/output (I/O) channel and the GPIB Adapter. The channel's data-bus signals, D0 through D7, are buffered by a Transceiver and become the internal data-bus signals, GD0 through GD7.

## Control-Signal Buffer

The control-signal buffer acts as a line receiver to minimize loading on the system's I/O channel and to increase noise immunity. The I/O channel's control signals, the DMA Controller's acknowledge signals, and the address lines all are received by a Line Receiver before being used on the GPIB Adapter. The channel signals, when received by the Line Receiver, are given different names to avoid confusion. The following table shows the channel signal names.

	PC I/O Signal Name	GPIB Adapter Name
Address Lines	A10 A11 A12	RS0 RS1 RS2
Control Signals	$\overline{\text{IOR}}$ IOW RESET DRV T/C	Read ( $\overline{\text{RD}}$ ) Write ( $\overline{\text{WR}}$ ) RESET GT/C
DMA Control Acknowledge Signals	$\overline{\text{DACK 1}}$ $\overline{\text{DACK 2}}$ $\overline{\text{DACK 3}}$	$\overline{\text{DACK}}$ $\overline{\text{DACK}}$ $\overline{\text{DACK}}$

## DMA Circuitry

The DMA circuitry recognizes when direct-memory access (DMA) operations are enabled or disabled, and routes the DMA request and acknowledge signals between the adapter and the selected DMA channels.

The DMA acknowledge and request signals at the channel connector are brought to jumper pin arrays on the board. These pin arrays are arranged so that two small pin-to-pin shunt connectors, which are supplied with the adapter, can be used to select the proper pair of DMA signals for the adapter's use. Side B of the pin array is connected to the channel connector, and side A is connected to 'DMA acknowledge' (DACK) and 'DMA request' (DRQ DLY).

$\overline{\text{DACK}}$  is ANDed with 'DMA enable' ( $\overline{\text{DMA EN}}$ ) and becomes 'DMA acknowledge' ( $\overline{\text{DMAACK}}$ ). When  $\overline{\text{DMAACK}}$  goes low, both halves of U5 (a dual 74LS74A flip-flop) are cleared, and the  $\overline{\text{DMAACK}}$  pin on the Talker/Listener/Controller (TLC) goes low. When  $\overline{\text{DMAACK}}$  returns to high, the  $\overline{\text{DMAACK}}$  pin on the TLC goes high. The flip-flops delay the low-to-high transition of the  $\overline{\text{DMAACK}}$  signal for a minimum of 200 nanoseconds to prevent any spurious TLC DMAREQ pulses from being sent to the DMA Controller.

## Interrupt Circuitry

The interrupt circuitry recognizes when interrupts have been enabled or disabled and passes or inhibits them accordingly.

The interrupt circuitry consists of logic and an array of jumper pins to allow user selection of the channel 'interrupt request' signal. The interrupt logic consists of one gate of a 74LS125A three-state buffer, which drives the A side of an array of jumper pins. The input of the buffer is tied to ground, and the enable for the gate is driven by the 'interrupt request' signal ( $\overline{\text{INT REQ}}$ ), which is generated by the shared interrupt logic. The B side of the jumper pin array is connected to the channel 'interrupt request' signals, IRQ2 through IRQ7. The pin array is arranged so that a small pin-to-pin shunt connector, which is supplied with the adapter, can be used to select the desired interrupt-request level.

Because the input of the three-state buffer is tied to ground, an active interrupt request ( $\overline{\text{INT REQ}}$  is low) corresponds to a low level being sent onto the selected interrupt-request bus line.

When  $\overline{\text{INT REQ}}$  is high, no interrupt is being requested, and the output of the buffer's gate is in a high-impedance state. The 8.2 kilo-ohm pull-up resistor attached to the output of the gate ensures that the IRQX signal (X is 2, 3, 4, 5, 6, or 7) is pulled to a high level when the board is not requesting an interrupt.

## Shared Interrupt Logic

The shared interrupt logic, when used with the appropriate interrupt-handling programs, allows multiple adapters to use or share the same system I/O channel's interrupt-request line.

This interrupt-sharing scheme works properly only when adapters using the same interrupt level are installed in the same unit.

The IRQX line is treated as an open-collector type of line, even though it is connected to the output of a three-state gate. This gate pulls the line to a low level when the adapter is requesting an interrupt. Adapters using the IRQX line also can pull the line to ground at the same time. Other adapters using the same IRQX line, but not requesting an interrupt, are not driving the line, because their gates are in a high-impedance state.

The shared interrupt logic generates a low-going pulse that is two system clock periods long to request an interrupt from the processor.

When the 'interrupt enable' signal (INT EN) from the interface control logic is high, the GPIB Adapter can request interrupts. Interrupt requests to the shared interrupt logic originate from the interrupt signal (INT) of the TLC (TLC INT) and the DMA terminal-count interrupt (DMA T/C INT) logic.

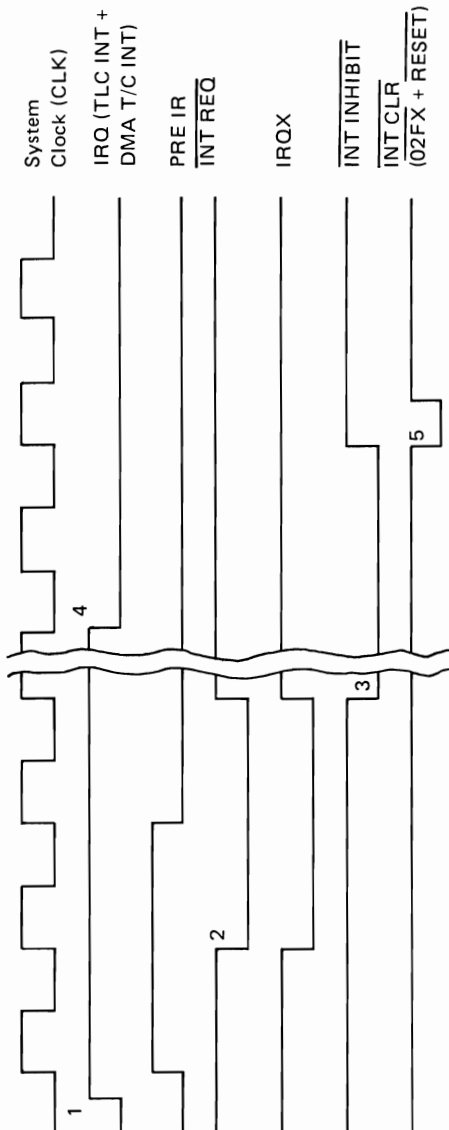
The TLC INT signal goes high, when the corresponding TLC mask bit is unmasked and any of the following GPIB events occur:

- Command pass through (unrecognized command) received
- Address pass through (unrecognized or secondary address) received
- Device trigger received
- End message received
- Device clear received
- Error (no listeners)
- Data out (ready to originate data byte)
- Data in (data byte accepted)
- Service request interrupt
- Command out (ready to originate command byte)
- Lockout-mode status change
- Remote-mode status change
- Address status change

The DMA T/C INT signal goes high whenever the DMA Controller channel being used by the adapter during a DMA transfer reaches terminal count.

If the interrupt handler chooses not to clear the source of the interrupt, the **IRQX** high-to-low-to-high pulse is regenerated after the handler writes to port hex 02FX. The shared interrupt logic will generate a continuous series of **IRQX** pulses, thereby causing a series of interrupts, until the handler services it.

The following is a timing diagram showing how the shared interrupt logic functions.



- 1 - Interrupt request from TLC INT or DMA T/C INT
- 2 - Selected PC interrupt line high-to-low transitions
- 3 - **INT INHIBIT** low prohibits any further **INT REQ** low transitions
- 4 - Interrupt handler invoked, source of interrupt cleared
- 5 - **RESET** or write to I/O port hex 02FX (X=2-7) reenables shared interrupt logic

## DMA T/C Interrupt

The DMA T/C interrupt logic determines when the DMA Controller has reached terminal count and generates an interrupt request.

The 'DMA acknowledge' signal ( $\overline{DACK}$ ) and the DMA T/C signals from the DMA Controller are received by the DMA circuitry and the control-signal buffer and become  $\overline{DMAACK}$  and GT/C, respectively. These signals and INT EN generate an interrupt request (DMA T/C INT) when the DMA Controller reaches terminal count (GT/C makes a low-to-high transition). DMA T/C INT goes high on a terminal-count condition only if interrupts were previously enabled by setting any one of the enable-interrupt bits in interrupt mask register 1 or 2 (IMR1 or IMR2).

The DMA T/C INT signal is cleared to a low state when 'reset' goes high or when interrupt status register 2 (ISR2) is read. An interrupt handler routine should read ISR2.

## Address Decode Logic

The address decode logic monitors the 16 address lines (A0 through A15) of the system's I/O channel to determine when the GPIB Adapter's I/O address is present on the channel, and enables read and write access to the adapter. A 13-input NAND gate is used, along with some inverters and jumpers, to decode the base address, 'TLC select' ( $\overline{\text{TLC SEL}}$ ). The 'address enable' signal (AEN) from the processor ensures that the adapter is not inadvertently selected when the system processor does not have control of the bus.

The binary base address decoded by the NAND gate is as follows.

PC Address Select			TLC Register Select			GPIB Adapter Base Address									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
x	x	x	y	y	y	1	0	1	1	1	0	0	0	0	1

x PC Address Select (Jumpers)

y TLC Register Select (Programmed)

A second NAND gate decodes the 'interrupt enable' signal,  $\overline{02FX}$ . The binary decode process is as follows.

Interrupt Enable Base Address													Interrupt Level		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	1	1	1	1	0	z	z	z

z Interrupt Request Level. Must correspond to the IRQ level selected by jumpers.

The  $\overline{\text{WR}}$  and  $\overline{\text{AEN}}$  signals ensure that the  $\overline{02FX}$  signal is generated only when the system processor is writing to the decoded address.



## $\mu$ PD7210 Talker/Listener/Controller (TLC)

The TLC implements almost all interface functions to interact with other devices on the bus. Within the TLC are 21 program registers, which are used to set up, control, and monitor the interface functions and to pass commands and data to and from the bus. Access to these functions is through 8 read-only registers and 8 write-only registers, of which 5 are indirectly addressed.

The TLC is enabled when the  $\overline{\text{TLC SEL}}$  signal is low and the 'register select' signals, RS0 through RS2, are decoded internally to gain access to the appropriate register. Data on the internal data bus (GD0 through GD7) is loaded into write-only registers at the trailing edge of  $\overline{\text{WR}}$ . Data in the read-only registers is placed on the internal data bus for a minimum access time after both  $\overline{\text{TLC SEL}}$  and  $\overline{\text{RD}}$  become low.

Most of the interface functions can be implemented or initiated from either side of the TLC. The distinction between the two is generally that between local and remote interface messages.

### Read Interface Registers

The following table lists the read-only interface registers and their corresponding bits.

Register		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Num.	Name	Name	Name	Name	Name	Name	Name	Name
DIR	+0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
ISR1	+1	CPT	APT	DET	ENDRX	DEC	ERR	DO	D1
ISR2	+2	INT	SRQI	LOK	REM	CO	LOKC	REMC	ADSC
SPSR	+3	S8	PEND	S6	S5	S4	S3	S2	S1
ADSR	+4	CIC	$\overline{\text{ATN}}$	SPMS	LPAS	TPAS	LA	TA	MJMN
CPTR	+5	CPT7	CPT5	CPT4	CPT4	CPT3	CPT2	CPT1	CPT0
ADR0	+6	X	DT0	DL0	AD5-0	AD4-0	AD3-0	AD2-0	AD1-0
ADR1	+7	EOI	DT1	DL1	AD5-1	AD4-1	AD3-1	AD2-1	AD1-1

X indicates bit is not used. Read bits that are not used may be read as 1 or 0.

## DIR (Data-In Register)

The data-in register receives data and commands from the bus.

## ISR1 (Interrupt Status Register 1)

The ISR1 records the occurrence of 8 conditions or events. This register is not a true status register because the bits are cleared whenever they are read.

The following table describes the bits of the ISR1 register.

Bit	Name	Value	Function
7	CPT	1	Command Pass Through. An undefined command has been received over the bus.
6	APT	1	Address Pass Through. The secondary address (Address Mode 3) has been received.
5	DET	1	Device Execute Trigger. The Device Execute Trigger (DET) command has been received.
4	END RX	1	End Received. EOI or EOS command has been received.
3	DEC	1	Device Clear. The Device Clear (DCL) command has been received.
2	ERR	1	Error. The contents of the Command/Data Out Register (CDOR) have been lost.
1	DO	1	Data Out. A data write request issued to the CDOR.
0	DI	1	Data In. A byte has been written to the DIR, and the DIR should be read.

## ISR2 (Interrupt Status Register 2)

The ISR2 records the occurrence of 8 conditions or events. This register is not a true status register because the bits are cleared whenever they are read.

The following table describes the bits of the ISR2 register.

Bit	Name	Value	Function
7	INT	1	Logical OR of the enabled interrupt status bits.
6	SRQI	1	Service Request In. A respond to service request (SRQ) message has been received while the controller is active.
5	LOK	1	Lockout (non-interrupt bit). The device is in local with lockout state (LWLS) or remote with lockout state (RWLS).
4	REM	1	Remote (non-interrupt bit). The device is in remote state (REMS) or RWLS.
3	CO	1	Command Output. A request for a command to be written to the CDOR.
2	LOKC	1	Lockout Change. The value of the LOK bit (bit 5) has changed.
1	REMC	1	Remote Change. The value of the REM bit (bit 4) has changed.
0	ADSC	1	Address Status Change. One of the four bits (TA, LA, CIC, MJMN) of the address status register has changed.

## SPSR (Serial Poll Status Register)

The serial poll status register echoes the contents of the serial poll mode register (SPMR). This status can be read to confirm that a request for a serial poll was accepted.

## ADSR (Address Status Register)

The address status register contains information about the current addressed state of the GPIB Adapter.

The following table describes the bits of the ADSR register.

Bit	Name	Value	Function
7	CIC	1	The device is controller in charge.
6	ATN	1	Attention. The device is in data transfer mode; the $\overline{\text{ATN}}$ line is high.
5	SPMS	1	Serial poll mode state: The serial poll enable (SPE) message has been received.
4	LPAS	1	The device is in listener primary addressed state.
3	TPAS	1	The device is in talker primary addressed state.
2	LA	1	Listener Addressed: The device is in listener addressed state.
1	TA	1	Talker Addressed: The device is in talker addressed state.
0	MJMN	1	A major-talk or major-listen address has been received.
		0	A minor-talk or minor-listen address has been received.

## **CPTR (Command Pass Through Register)**

The command pass through register reads the data on the data input/output (DIO) lines for the following situations:

If ISR1 bit 7 (CPT) is equal to 1 and auxiliary register B (AUXRB) bit 0 (CPT ENABLE) is equal to 1 then CPTR has an undefined command or has received a secondary command after an undefined primary command.

If ISR1 bit 6 (APT) is equal to 1 and Address Mode 3 is selected then CPTR contains a secondary address.

After a parallel poll the CPTR contains the parallel-poll response message.

## ADR0 (Address Register 0)

Address register 0 contains the major address set by the address register (ADR), as well as the functions enabled for that address.

The following table describes the bits of the ADR0 register.

Bit	Name	Function
7		Not used
6	DT0	Disable Talker 0. Set or cleared by a write to ADR.
5	DL0	Disable Listener 0. Set or cleared by a write to ADR.
4-0	AD5-0 to AD1-0	Major Address. Set by a write to ADR.

## ADR1 (Address Register 1)

Address register 1 contains the minor address set by the address register (ADR), as well as the functions enabled for that address.

The following table describes the bits of the ADR1 register.

Bit	Name	Function
7	EOI	End or Identify. Indicates that EOI message was sent on the last data byte received.
6	DT1	Disable Talker 1. Set or cleared by a write to ADR.
5	DL1	Disable Listener 1. Set or cleared by a write to ADR.
4-0	AD5-1 to AD1-1	Minor Address. Set by a write to ADR.

## Write Interface Registers

The following table lists the write-only interface registers.

Register		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Num.	Name	Name	Name	Name	Name	Name	Name	Name
CDOR	+0	CDO7	CDO6	CDO5	CDO4	CDO3	CDO2	CDO1	CDO0
IMR1	+1	CPTIE	APTIE	DETIE	ENDIE	DECIE	ERRIE	DOIE	DIIE
IMR2	+2	X	SRQIE	DMAO	DMAI	COIE	LOKIE	REMCIE	ADSCIE
SPMR	+3	S8	rsv	S6	S5	S4	S3	S2	S1
ADMR	+4	ton	lon	TRM1	TRM0	X	X	ADM1	ADM0
AUXMR	+5	CNT2	CNT1	CNT0	COM4	COM3	COM2	COM1	COM0
ADR	+6	ARS	DT	DL	AD5	AD4	AD3	AD2	AD1
EQSR	+7	EC7	EC6	EC5	EC4	EC3	EC2	EC1	EC0

X indicates bit is not used.

### CDOR (Command/Data Out Register)

The command/data out register is an output-only register used to send commands or data to the bus.

### IMR1 (Interrupt Mask Register 1)

The interrupt mask registers are used to enable or disable the generation of the INT signal on the occurrence of key events.

The following table describes the bits of the IMR1 register.

Bit	Name	Function
7	CPTIE	Command Pass Through, Interrupt Enable
6	APTIE	Address Pass Through, Interrupt Enable
5	DETIE	Device Trigger, Interrupt Enable
4	ENDIE	END message (EOI) or EOS message received, Interrupt Enable
3	DECIE	Device Clear, Interrupt Enable
2	ERRIE	Error, Interrupt Enable
1	DOIE	Data Out, Interrupt Enable
0	DIIE	Data In, Interrupt Enable

## IMR2 (Interrupt Mask Register 2)

The following table describes the bits of the IMR2 register.

Bit	Name	Function
7		Write a 0 to this bit.
6	SRQIIE	Service Request In, Interrupt Enable
5	DMAO	DMA Output (non-interrupt). Enables or disables a DMA transfer to the data registers
4	DMAI	DMA Input (non-interrupt). Enables or disables a DMA transfer to the data registers.
3	COIE	Command Output, Interrupt Enable
2	LOKCIE	Lockout Change, Interrupt Enable
1	REMCIE	Remote Change, Interrupt Enable
0	ADSCIE	Address Status Change, Interrupt Enable

## SPMR (Serial Poll Mode Register)

The serial poll mode register holds the status byte and the local request service (rsv) message (bit 6). When rsv is 1, the adapter enters the service request state (SRQS). When a serial poll is requested, the adapter sends the contents of the SPMR over the bus and clears the rsv bit upon completion of the poll.



## ADMR (Address Mode Register)

The address mode register selects the functions of the Transmit and Receive pins (T/R2 and T/R3) and selects the address mode.

The following table shows the functions of pins T/R2 and T/R3 for the various settings of bits 4 and 5.

Bit 5 TRM1	Bit 4 TRM0	T/R2	T/R3
0	0	EOIOE	TRIG
0	1	CIC	TRIG
1	0	CIC	EOIOE
1	1	CIC	PE

The following table shows the various address modes as selected by bits 0, 1, 6, and 7.

Bit 7 ton	Bit 6 lon	Bit 1 ADM1	Bit 0 ADM0	Address Mode	Contents of Adr. Reg. 0	Contents of Adr. Reg. 1
1	0	0	0	Talk Only	Not Used	Not Used
0	1	0	0	Listen Only	Not Used	Not Used
0	0	0	1	Address Mode 1	Major Talk or Major Listen Address	Minor Talk or Minor Listen Address
0	0	1	0	Address Mode 2	Primary Address Talk or Listen	Secondary Address Talk or Listen
0	0	1	1	Address Mode 3	Primary Address Major Talk or Major Listen	Primary Address Minor Talk or Minor Listen

## AUXMR (Auxiliary Mode Register)

A write to the auxiliary mode register causes one of the following to occur:

- An auxiliary command is issued.
- The state-change-prohibit-time is set using the internal counter register.
- The parallel poll register is written to.
- Auxiliary register A, B, or E is written to.

The following table shows the 5 internal registers and their corresponding functions.

Control Code			Command Code				
CNT2	CNT1	CNT0	COM4	COM3	COM2	COM1	COM0
When CNT2-CNT0 is:			ICR is loaded with:				
0	0	1	0	CLK3	CLK2	CLK1	CLK0
			PPR is loaded with:				
0	1	1	U	S	P3	P2	P1
			AUXRA is loaded with:				
1	0	0	BIN	XEOS	REOS	HLDE	HLDA
			AUXRB is loaded with:				
1	0	1	ISS	INV	TRI	SPEOI	CPT ENABLE
			AUXRE is loaded with:				
1	1	0	0	0	0	DHDC	DHDT

To issue the auxiliary commands shown in the following table, write a binary 000COM<sub>4</sub>COM<sub>3</sub>COM<sub>2</sub>COM<sub>1</sub>COM<sub>0</sub> to the auxiliary mode register.

The following table lists the auxiliary command summary.

Auxiliary Command	Function Code*					Hex Code**
	COM4	COM3	COM2	COM1	COM0	
Immediate Execute pon	0	0	0	0	0	00
Chip Reset	0	0	0	0	1	02
Finish Handshake	0	0	0	1	1	03
Trigger	0	0	1	0	0	04
Return to Local	0	0	1	0	1	05
Send EOI	0	0	1	1	0	06
Non-Valid Secondary Command or Address	0	0	1	1	1	07
Valid Secondary Command or Address	0	1	1	1	1	0F
Clear Parallel Poll Flag	0	0	0	0	1	01
Set Parallel Poll Flag	0	1	0	0	1	09
Take Control Asynchronously (pulsed)	1	0	0	0	1	11
Take Control Synchronously	1	0	0	1	0	12
Take Control Synchronously on End	1	1	0	1	0	1A
Go to Standby	1	0	0	0	0	10
Listen	1	0	0	1	1	13
Listen in Continuous Mode	1	1	0	1	1	1B
Local Unlisten	1	1	1	0	0	1C
Execute Parallel Poll	1	1	1	0	1	1D
Set IFC	1	1	1	1	0	1E
Clear IFC	1	0	1	1	0	16
Set REN	1	1	1	1	1	1F
Disable System Control	1	0	1	0	0	14

\*CNT2-CNT0 set to 000 binary

\*\*Represents all eight bits of the auxilliary mode register

## AUXMR Internal Registers

The following information describes the AUXMR internal registers.

### Internal Counter Register (ICR):

A write to this register (binary 0010clk<sub>3</sub>clk<sub>2</sub>clk<sub>1</sub>clk<sub>0</sub>) sets the state-change-prohibit-times T<sub>1</sub>, T<sub>6</sub>, T<sub>7</sub>, and T<sub>9</sub> as referenced in the ANSI/IEEE std.488-1978.

The ICR should be set to equal the system clock frequency. The following table shows the internal counter register.

ICR							
7	6	5	4	3	2	1	0
0	0	1	0	clk <sub>3</sub>	clk <sub>2</sub>	clk <sub>1</sub>	clk <sub>0</sub>

## Parallel Poll Register (PPR):

You can write to the parallel poll register by writing a binary 011USP<sub>3</sub>P<sub>2</sub>P<sub>1</sub> to the AUXMR.

You should not write to this register if you are using subset PP1 (remote parallel poll configuration) as the parallel poll (PP) interface function.

The following table shows the functions of the bits of the parallel poll register.

Bit	Bit Name	Value	Function
4	U	1	Disables participation in parallel poll
		0	Enables participation in parallel poll
3	S	1	Sense of status is the same as the ist message (If S=ist=1, PPR is logical 1).
		0	Sense of status is reversed from the ist (If S=0 and ist=0, PPR is a logical 1).
2-0	P(3)-P(1)		Binary value of which data line (DIO8-DIO1) to assert during a parallel poll.

**U = Parallel Poll Unconfigure;**

**S = Status Bit Polarity**

## Auxiliary Register A (AUXRA):

You can write to this register by writing a binary  $100A_4A_3A_2A_1A_0$  to the AUXMR.

The data-receiving modes are set by bits 1 ( $A_1$ ) and 0 ( $A_0$ ) as follows.

$A_1$ HLDE	$A_0$ HLDA	Data Receiving Mode
0	0	Normal Handshake Mode
0	1	Ready for Data (RFD) Hold Off on All (HLDA) Data Mode
1	0	RFD Hold Off on End (HLDE) Mode
1	1	Continuous Mode

The transmission and receiving modes of the EOI or EOS message are set by bits 4 ( $A_4$ ), 3 ( $A_3$ ), and 2 ( $A_2$ ) as follows.

Bit Name	Value	Function	Description
$A_4$ BIN	0 1	7-bit EOS 8-bit EOS	Selects seven or eight bits as the valid length of the EOS message
$A_3$ XEOS	0 1	Prohibit Permit	Transmit EOS. Permits or prohibits the automatic transmission of the EOI message at the same time as the EOS message in talker-active-state (TACS).
$A_2$ REOS	0 1	Prohibit Permit	Receive EOS. Permits or prohibits setting the EOI bit at reception of the EOS message

## Auxiliary Register B (AUXRB):

You can write to this register by writing a binary  $101B_4B_3B_2B_1B_0$  to the AUXMR.

Following is a description of the bits of the AUXRB register.

Bit Name	Value	Function	Description
$B_4$ ISS	0	Individual status (ist)=Parallel Poll flag	The value of the Parallel Poll flag is taken as the ist local message
	1	ist=SRQS	SRQS indicates the value of the ist local message (the Parallel Poll flag is ignored) SRQS=ist=1; SRQS=ist=0
$B_3$ INV		1 INT 0 INT	Specifies the active level of the INT pin
$B_2$ TRI	0	T(1) (low speed)	Sets low speed as T(1) in all cases
	1	T(1) (high speed)	Sets high speed as T(1) of handshake after transmission of second byte following data transmission
$B_1$ SPEOI	1 0	Permit Prohibit	Permits or prohibits the transmission of the END message in serial poll address state.
$B_0$ CPT ENABLE	1 0	Permit Prohibit	Permits or prohibits the setting of the CPT bit on receipt of an undefined command

## Auxiliary Register E (AUXRE):

You can write to auxiliary register E by writing a binary 110000E<sub>1</sub>E<sub>0</sub> to the AUXMR.

Following is a description of the bits of the AUXRE register.

Bit Name	Value	Function
E <sub>1</sub> DHDC	1 enables 0 disables	Enables or disables DAC hold-off by initiating device clear active state (DCAS)
E <sub>0</sub> DHDT	1 enables 0 disables	Enables or disables DAC hold-off by initiating device trigger active state (DTAS)



## ADR (Address Register)

A write to the address register sets the addresses and functions for ADR0 (major) and ADR1 (minor).

Following is a description of the bits of the ADR register.

Bit	Bit Name	Value	Function	Description
7	ARS	0 1	Address register 0 Address register 1	Selects the address register (ADR0 or ADR1) to which the low-order bits are written (AD1 to AD5)
6	DT	0 1	Permitted Prohibited	Permits or prohibits the set address (AD1 to AD5) detected as a talk address. This bit corresponds to DT1 or DT0 of the address registers.
5	DL	0 1	Permitted Prohibited	Permits or prohibits the set address (AD1 to AD5) detected as a listen address. This bit corresponds to DL1 or DL0 of the address registers.
4-0	AD1 to AD5			These bits indicate device addresses and correspond to AD5-0 to AD1-0 and AD5-1 to AD1-1.

## EOSR (End-of-String Register)

The end-of-string register contains the 7- or 8-bit EOS message byte used by the GPIB Adapter to detect the end of a data block transfer.

## **GPIB Transceivers**

Special-purpose, multi-function, GPIB transceivers serve as the interface between the adapter and the bus.

The TLC communicates with the bus through two special-purpose transceivers, a DS75160A for the data signals, and a DS75162A for the handshake and interface management signals. The direction of signals through these transceivers is controlled by three signals from the TLC (TE, DC, and PE) and the 'system control' (SC) signal from the interface control logic.

'Talk enable' (TE) is high when the TLC is a talker or active controller, and low when it is a listener. It controls the direction of the data, handshake, and EOI signals.

'Direction control' (DC) is inverted so that it is high when the TLC is controller-in-charge (CIC), and low at other times. It controls the direction of the ATN and SRQ signals.

'Pull-up enable' (PE) is high when the three-state driver mode is active, and low when the open collector mode is active. When a parallel poll is requested, the transceiver switches to open collector mode.

SC controls the direction of the 'interface clear' (IFC) and 'remote enable' (REN) signals, driving the bus when SC is high and receiving from the bus when it is low.

## Interface Control Logic

The interface control logic monitors the GPIB Adapter's data and control busses and generates signals that control interrupt requests, DMA requests, and system-controller capability.

The PAL12L6 is the main component in this group. The PAL monitors the internal data bus (GD0 through GD7), the 'TLC select' signal ( $\overline{\text{TLC SEL}}$ ), and the 'TLC register select' signals (RS0 through RS3). These inputs allow the internal logic of the PAL to generate the following outputs:

- $\overline{\text{SC OFF}}$ , or 'system controller off', which is true (low) when the TLC AUXMR is being selected and the Chip Reset or Disable System Control auxiliary commands are being sent.
- $\overline{\text{SC ON}}$ , or 'system controller on,' which is true (low) when the command being written to the AUXMR involves setting or clearing the GPIB IFC or REN signals.
- $\overline{\text{IR2}}$ , or 'TLC interrupt register 2,' which is true (low) when either ISR2 or IMR2 is being selected by the 'register select' lines.
- $\overline{\text{DMA ON}}$ , which is true (low) when the internal data bus indicates that the DMA out (DMAO) or DMA in (DMAI) bits of IMR2 are being set.
- $\overline{\text{IR1}}$ , or 'TLC interrupt register 1,' which is true (low) when either ISR1 or IMR1 is being selected by the 'register select' lines.
- INT ON, or 'interrupt on,' which is true (high) when the internal data bus signifies that any of the interrupt-enable bits in IMR1 or IMR2 are being set.

The SC signal is latched high, signifying that the GPIB Adapter is the system controller, whenever the TLC receives an auxiliary command to set or clear the REN or IFC signals. SC is returned to a low level when a Disable System Control or a Chip Reset auxiliary command is sent to the TLC, or when  $\overline{\text{RESET}}$  is low.

The  $\overline{\text{DMA EN}}$  signal is latched at a low level when either of the DMA enable bits in the TLC IMR2 is set.  $\overline{\text{DMA EN}}$  is returned to a high level if both bits are cleared or if  $\overline{\text{RESET}}$  is low.

The INT EN signal is latched at a high level if any of the interrupt-enable bits in IMR1 or IMR2 are set. INT EN is returned to a low level if  $\overline{\text{RESET}}$  is low or if all interrupt-enable bits are cleared.

# Programming Considerations

When power is switched on, the system issues a bus reset causing the following:

- The system-controller bit is cleared.
- The 'interrupt request' line is tri-stated.
- 'DMA request' is tri-stated.
- RESET is sent to the TLC, causing the following:
  - The local command, power on (pon), is set, and the interface functions are placed into their idle states.
  - The serial poll mode register (SPMR) is cleared.
  - The end-or-identify (EOI) bit is cleared.
  - AUXRA, AUXRB, and AUXRE are cleared.
  - The Parallel Poll flag and the request system control (rsc) locate message are cleared.
  - The transmit- and receive-mode bits (TRM0 and TRM1) in the address mode register are cleared.
  - All auxiliary mode commands are cleared and prevented from executing.

All other register contents should be considered as undefined after a bus reset has been issued.

The GPIB Adapter's other registers can be initialized while pon is set. A typical programmed initialization sequence for the adapter may include the following steps:

1. Write the Chip Reset auxiliary command (hex 02) to AUXMR.
2. Set or clear the desired interrupt-enable bits in IMR1 and IMR2.
3. Write to ADR to set the desired address for both ADR0 and ADR1.
4. Set the TRM0 and TRM1 bits and select the desired addressing mode in the ADMR.
5. Write the serial poll response to the SPMR.
6. If using a remote setup, clear the parallel poll register (PPR). If using a local setup, load the parallel poll response into the PPR.
7. Clear pon by issuing the Immediate Execute pon auxiliary command.

The auxiliary commands can now be executed.

## The GPIB Adapter as a Controller

The GPIB Adapter can become controller-in-charge (CIC) either by being the system controller and taking control or by having control of the bus passed from the current active controller.

The active controller has the ability to make ATN active and send multi-line commands and conduct parallel polls.

If the GPIB Adapter is the system controller, it can take control by executing the following command sequence:

1. Issue the Set IFC (hex 1F) auxiliary command. (This command should never be executed if the adapter is not system controller.)
2. Wait a minimum of 100 microseconds
3. Issue the Clear IFC (hex 1E) auxiliary command.

If the bus has an active controller, the GPIB Adapter waits until the controller becomes idle before making ATN active.

If the adapter is not the system controller, it can be passed control by the active controller using the following sequence:

1. Receive the My-Talk-Address (MTA) command.
2. Receive the Take Control (TCT) command.
3. The active controller sees the completed handshake and makes ATN inactive.
4. The adapter automatically becomes CIC and makes ATN active.

When the adapter becomes the active controller, the following bits are set:

- System controller bit
- CIC bit in the ADSR
- CO bit in ISR2.

The GPIB Adapter sends commands as an active controller by writing to the CDOR in response to the setting of the CO bit in ISR2. The TLC responds to and recognizes the interface commands it sends as well as receives.

To perform data transfers, the GPIB Adapter enters the controller standby state (CSBS). When the data transfer is complete, the adapter resumes active control of the bus. The way this is done depends on the role the adapter is to play in the data transfer. The following three cases describe the ways the adapter completes a data transfer and then resumes control.

#### Case 1: GPIB Adapter as the Talker

1. If the CO bit is set, send the GPIB Adapter's MTA.
2. With CO set, issue the Go to Standby auxiliary command (hex 10).
3. Complete the data transfer.
4. If DO is set, issue the Take Control Asynchronously auxiliary command (hex 11).



## Case 2: GPIB Adapter as the Listener

1. With CO set, issue the Listen auxiliary command (hex 13).
2. With CO set, issue the Go to Standby auxiliary command (hex 10).
3. Begin data transfer.
4. After the DI bit is set and before reading the last byte from the DIR, issue the Take Control Synchronously auxiliary command (hex 12).
5. Read the DIR. The GPIB Adapter then becomes the active controller automatically.

**Note:** The last byte of the data transfer can be detected by checking the END RX bit in ISR1.

## Case 3: GPIB Adapter as neither the Talker nor Listener

**Note:** The talker must finish the data transfer with the END or EOS message.

1. With CO set, issue the Listen auxiliary command (hex 13).
2. Set the hold-off-on-end (HLDE) bit in AUXRA.
3. Issue the Go to Standby auxiliary command (hex 10).
4. Begin data transfer.
5. When DI is set, read the DIR so that the handshake cycle can be completed by other devices on the GPIB.
6. When HLDE occurs, issue the Take Control Synchronously auxiliary command. Hold-off also can be detected by reading the END RX bit in ISR1.

In Cases 2 and 3, the Take Control Asynchronously auxiliary command can be issued when the possibility of disrupting an in-progress handshake is acceptable.

In all cases, a CO status indicates the GPIB Adapter is now the active controller.

The GPIB Adapter can relinquish control of the bus and become an idle controller in two ways:

1. By issuing the Chip Reset or Disable System Control auxiliary command.
2. By passing control to another device. This is done by sending the MTA of the other device followed by the TCT interface command. The GPIB Adapter makes ATN inactive as soon as the TCT command is accepted.

# The GPIB Adapter as a Talker or Listener

The GPIB Adapter may be either the GPIB talker or listener, but not both simultaneously. Either function is deactivated automatically if the other is activated. The talker-addressed (TA), listener-addressed (LA), and  $\overline{\text{ATN}}$  bits in the ADSR indicate the specific state of the adapter as follows:

$\overline{\text{ATN}}$	TA	LA	
0	1	0	Addressed Talker—cannot send data
1	1	0	Active Talker—can send data
0	0	1	Addressed Listener—cannot receive data
1	0	1	Active Listener—can receive data

## Programmed Implementation

When the GPIB has no controller, the talk only (ton) and listen only (lon) local commands are used to set the talker and listener interface functions. These commands are set in the ADMR and should be programmed during initialization.

## Addressed Implementation

The GPIB Adapter responds to the address command that the active controller sends over the bus, regardless of which device is in control. This enables the adapter to respond to its own addressing commands which the adapter sends over the bus. The adapter has three address modes, which are selected by writing to the ADMR. Descriptions of the address modes follow.

## Address Mode 1

In this mode, the GPIB Adapter responds to two primary addresses, ADR0 (major) and ADR1 (minor).

The receipt of the GPIB Adapter's primary My Listen Address (MLA) command has the following effects:

- LA bit in ADSR is set.
- ADSC bit in ISR2 is set.
- MJMN bit in ADSR is set if MLA corresponds to ADR1.
- DI bit is set in ISR1 upon receipt of a data byte in the DIR.

The receipt of the GPIB Adapter's primary My Talk Address (MTA) command has the following effects:

- TA bit in ADSR is set.
- ADSC bit in ISR2 is set.
- MJMN bit in ADSR is set if MTA corresponds to ADR1.
- DO bit in ISR1 is set when the CDOR is ready to send another data byte.

## Address Mode 2

Address mode 2 is used for primary and secondary addressing as specified by the talker-extended (TE) and listener-extended (LE) interface functions. ADR0 contains the primary address, and ADR1 the secondary address.

The sequence for programming the TE interface function is shown in the following.

Step	Results
1. MTA received (Hex 40 + ADR0 address)	TPAS bit (in ADSR) = 1
2. MSA received (Hex 60 + ADR1 address)	TA in ADSR = 1 ADSC in ISR2 = 1 DO in ISR1 = 1 Now in TADS (Talker Addressed State)

The following shows the sequence for programming the LE interface function.

Step	Results
1. MLA received (Hex 20 + ADR0 address)	LPAS bit (ADSR) = 1
2. MSA received (Hex 60 + ADR1 address)	TA bit in ADSR = 1 ADSC bit in ISR2 = 1 DI in ISR1 = 1 (when data byte is available in DIR) Now in LADS (Listener Addressed State)

In both listener and talker addressing sequences, the My-Secondary-Address (MSA) command must be received before the receipt of another primary-command-group (PCG) command. The MJMN bit indicates which register is referred to by the address status.

## Address Mode 3

This mode is used to implement the TE and LE interface functions with the addition of two possible primary addresses, major and minor. The proper operation of address mode 3 is as follows:

Step	Results
1. Primary Address (MLA or MTA) received.	TPAS in ADSR = 1 if MTA received LPAS in ADSR = 1 if MLA received MJMN in ADSR = 1 if address is ADR1
2. Secondary Address received	APT in ISR1 = 1 Data Accepted handshake hold-off is activated
3. In program, determine if primary address is talk, listen, major, or minor (use TPAS, LPAS, and MJMN bits)	
4. Read the secondary address command from the CPTR, determine if it is the address of the GPIB Adapter	
5. If not the adapter's address, issue the non-valid (hex 07) auxilliary command.	Command was other secondary address (OSA), GPIB adapter enters Listen and Talk idle state. Handshake is completed.
6. If the secondary address is the adapter's address, issue the valid (hex 0F) auxilliary command.	LA = 1 and TA = 0 if LPAS = 1 TA = 1 and LA = 0 if TPAS = 1 Data-accepted message sent true and handshake is completed. Now in TADS (TA=1) or LADS (LA=1).

In both listen- and talk-mode sequences, the secondary address command will continue to generate a 'data accepted' holdoff with APT set until another PCG command is received. In this way, the controller can address several devices having the same primary address without repeating the primary address command.

# **Sending and Receiving Commands**

When the GPIB Adapter is a talker or listener, data or device-dependent commands can be sent or received using DMA or programmed I/O.

## **Programmed I/O Data Transfer**

When the GPIB Adapter is operating as a talker, the DO bit in ISR1 is set when all addressed listeners are ready to receive a data byte on the bus. The GPIB Adapter responds by writing the next byte to the CDOR.

When the GPIB Adapter is a listener, the DI bit in ISR1 is set when a data byte is available in the DIR. The DIR is then read to complete the handshake.

In both cases, remember that the DO and DI bits are cleared when ISR1 is read.

## **DMA Data Transfer**

When the GPIB Adapter is properly set up for a DMA transfer, the sending and receiving of data is controlled by the TLC and the DMA Controller. During the sending of data (DMAO bit is set), DRQ is driven high under the same conditions that set the DO bit. During the receipt of data (DMAI is set), DRQ is driven high under the same conditions that set the DI bit.

## **Sending END or EOS**

The END command is sent by issuing the Send EOI auxiliary command just before writing the last data byte to the CDOR. The EOS command is sent simply by making the last byte the EOS code.

## **Stopping on an END or EOS**

The END status bit in ISR1 is used to inform the program of the receipt of an END command or EOS command.



# Serial Polls

Serial polls allow the controller-in-charge to obtain detailed status information about each device set up for responding.

## Conducting Serial Polls

To conduct a serial poll, the adapter must:

1. Become active controller and send the Unlisten (UNL) command.
2. Send the My-Talk-Address (MTA) command of the device to be polled.
3. Send the Serial Poll Enable (SPE) command.
4. Issue the Listen auxiliary command.
5. Issue the Go to Standby auxiliary command.
6. Issue the Take Control Synchronously auxiliary command.
7. Read the device's status byte from the Data Input Register (DIR).
8. Send the Serial Poll Disable (SPD) command.
9. Repeat Steps 1 through 8 for all devices on the bus that must be polled.

## Responding to a Serial Poll

The following are recommended steps for requesting service:

1. Check the PENDING bit of the SPMR. If the bit is set, the GPIB Adapter is currently in the serial poll active state (SPAS).
2. If PENDING is 0, write to the SPMR the desired status byte (STB) with rsv (bit 6).
3. When the active controller polls the GPIB Adapter serially, the adapter automatically transfers the STB message to the bus with DIO7 made active, and then makes the SRQ line inactive.

**Note:** If a serial poll is in progress, the GPIB Adapter waits for its completion before making the SRQ line active.

If the serial poll end-of-information (SPEOI) bit of AUXRB is set, the EOI line is made active with the STB.

## Parallel Polls

The active controller uses parallel polls to check the status of several devices simultaneously. The meaning of the status returned by the polled devices is device-dependent, but two general uses for parallel polls follow:

- When the controller sees an active SRQ in a system with several devices, it can quickly determine which one requested a serial poll using, usually, only one parallel poll.
- In systems for which the response time required for the controller to service a device is short, and the number of devices is few, parallel polls can replace serial polls entirely, if the controller polls frequently.

## Configuring for a Parallel Poll

Before conducting a parallel poll, the controller must establish the desired parallel poll response for each device. This can be done in two ways:

- Local configuration (parallel poll function, subset PP2): This involves assigning the sense of the line and the 'response' line from the device side. This setup does not change once the device is installed.
- Remote configuration (parallel poll function, subset PP1): The active controller dynamically assigns the sense of the line and the 'response' line using interface commands. The GPIB Adapter, when operating as the system controller, should execute the following sequence:
  1. Become an active controller.
  2. Send the Unlisten (UNL) command.
  3. Send the My-Listen-Address (MLA) of the first device to be configured.
  4. Send the Parallel Poll Configure (PPC) command.
  5. Send the Parallel Poll Enable (PPE) command for that device.
  6. Repeat Steps 2 through 5 for each additional device.

Because there are 8 data lines, each with two possible responses (true or false), 16 responses are possible. The data lines are driven open collector during parallel polls, so more than one device can respond on a certain data line. A device sending the line true overrides any device sending the line false.

For each data line, the sense for a parallel poll can be set as follows:

<b>DIO Line Value</b>	<b>Sense (S)</b>	<b>Individual Status (ist)</b>
True (1)	0	0
False (0)	0	1
False (0)	1	0
True (1)	1	1

## Conducting a Parallel Poll

To conduct a parallel poll, the GPIB Adapter does the following:

1. Becomes active controller.
2. Issues the Execute Parallel Poll auxiliary command (hex 1D).
3. Waits for the DO bit to be set, signaling the completion of the parallel poll.
4. Reads parallel poll response through the CPTR, and responds as necessary.

**Note:** If more than one device is driving a certain data line, Steps 2 through 4 may have to be repeated.

## Disabling Parallel Polling

To disable a device from participating in parallel polls, the GPIB Adapter:

1. Becomes active controller.
2. Sends the Unlisten (UNL) command.
3. Sends the My-Listen-Address (MLA) command for the device to be disabled.
4. Sends the Parallel Poll Disable (PPD) command.
5. Sends the Parallel Poll Unconfigure (PPU) command.
6. Repeats Steps 2 through 5 for each device to be disabled.

## Responding to a Parallel Poll

Before the GPIB Adapter can be polled by the CIC, the following must occur:

- For a remote setup, the parallel poll commands are interpreted without program assistance.
- For a local setup, the controller sets the desired parallel poll response by writing to the parallel poll register.
- The GPIB Adapter then selects the source and value of the local individual status (ist) message (see the following figure).

Parallel Poll Function	Individual Status Select (ISS in AUXRB)
ist set and cleared by the Set and Clear Parallel Poll Flag auxiliary commands.	0
ist and SRQ set when rsv set. ist, rsv, and SRQ cleared by completion of a parallel poll.	1

After proper setup, programming is required for setting and clearing the local ist command as desired.

## Interrupts

Interrupts are enabled through a hardware jumper to one of the six available IRQ lines on the system's I/O channel.

Interrupt requests from the TLC are enabled using the IE bits in IMR1 and IMR2.

The DMA T/C interrupt is implemented external to the TLC and is enabled whenever the DMAO or DMAI bit, and at least one of the IE bits, is set. The DMA T/C interrupt is discussed further in the next section.

The GPIB Adapter drives the selected IRQ line when at least one of the IE bits is set in IMR1 or IMR2; otherwise, the selected IRQ line is tri-stated.

Once made active, the interrupt request line remains active until the corresponding status register is read (ISR1 or ISR2).

The DMA T/C interrupt has no corresponding status register or bit, but is cleared when ISR2 is read.

Interrupts are selected by the Interrupt Controller on the system board. When none of the IE bits in the TLC is set, the GPIB Adapter's IRQ line is pulled to a high state. The IRQ line must be pulled high because a tri-state may look like an interrupt request to the Interrupt Controller.



An interrupt handler routine for the GPIB Adapter must do the following:

1. Read ISR2 to see if the INT bit is set, thus confirming that the GPIB Adapter has issued an interrupt. If the DMA T/C interrupt has been enabled, the INT bit will not be set when it occurs; however, the DMA Controller will indicate whether it has reached terminal count for the GPIB Adapter's DMA channel.
2. Read ISR1. Reading both ISR2 and ISR1 will clear an interrupt caused by any of the 13 possible conditions.
3. Write the End of Interrupt command to the Interrupt Controller.
4. Write to I/O address hex 2FX, where X is the interrupt level being used by the GPIB Adapter. Writing to 2FX reenables interrupts on the adapter.

In a system where several GPIB Adapters share the same interrupt level, Steps 1 through 4 should be used in each adapter's interrupt handler routine. Each of the adapters is polled until the one that caused the interrupt is found. When hex 2FX is written to, any pending interrupt from another GPIB Adapter is allowed to occur, and that adapter may then be serviced.

**Note:** Remember that the status bits in ISR1 and ISR2 automatically clear when the register is read. A software copy of these registers should be maintained.

## Programming the Interrupt Controller

Programming information for the Interrupt Controller chip may be found in Intel's *Intel Component Data Catalog*. Additional information may also be found in technical references for your system.

**Note:** At power-up time, the Interrupt Controller is initialized by the IBM PC BIOS program, which is in ROM on the system board. Programs written for processing GPIB Adapter interrupts must in no way change the overall configuration of the controller. Commands written to the controller should affect only the selected IRQ line.

The setup of the Interrupt Controller and the way it is used by the BIOS may be found in the BIOS program listing in your system's technical reference.

## DMA Transfers

DMA transfers must be enabled through two hardware jumpers to one of the three available pairs of DMA transfer lines on the system's I/O channel.

DMA requests from the TLC are enabled using the DMAO and DMAI bits in IMR2. The TLC generates a DMA request under the same conditions that set the DO and DI bits in ISR1. A DMA request indicates that the TLC requires either a byte to be written to the CDOR or a byte to be read from the DIR. The 'DMA request' signal (DRQ) is cleared by a low on the  $\overline{DACK}$ .

The GPIB Adapter drives the selected DRQ line and enables the DACK line whenever the DMAO or DMAI bit is set in IMR2. Otherwise, the selected DRQ line is tri-stated, and the DACK lines are disabled.

DMA transfers are selected by the DMA Controller on the system board. The DMA Controller should never be enabled to respond to a DMA request from the GPIB Adapter when neither the DMAO nor DMAI bit is set in the TLC. This tri-states the adapter's DRQ line, which may look like a DMA request from the adapter.

Once made active, the DMA request line remains active until a DMA transfer occurs, or until a read from the DIR or a write to the CDOR occurs, depending on the direction of the DMA transfer.

The DMA terminal count (T/C) interrupt is enabled whenever the DMAO or DMAI bit is set and at least one of the interrupts internal to the TLC is enabled.

The DMA T/C interrupt is asserted when the DMAO or DMAI bit is set, and the system's I/O channel T/C line sends a high pulse during a DMA transfer to the adapter. A high pulse on the T/C line means the DMA Controller chip has reached terminal count (that is, the DMA Controller's byte count has gone from 0 to FFFF for the corresponding DMA channel). The DMA T/C interrupt is cleared when ISR2 is read.

The DMA T/C interrupt can be detected by reading the status register or channel byte-count register of the DMA Controller. The terminal-count bit corresponding to the GPIB Adapter's selected DMA channel is set in the DMA Controller's status register when the controller reaches terminal count for that DMA channel. All terminal-count bits are cleared when the controller's status register is read. The channel byte count should be FFFF unless the DMA Controller has been programmed to start that channel automatically.

## Programming the DMA Controller

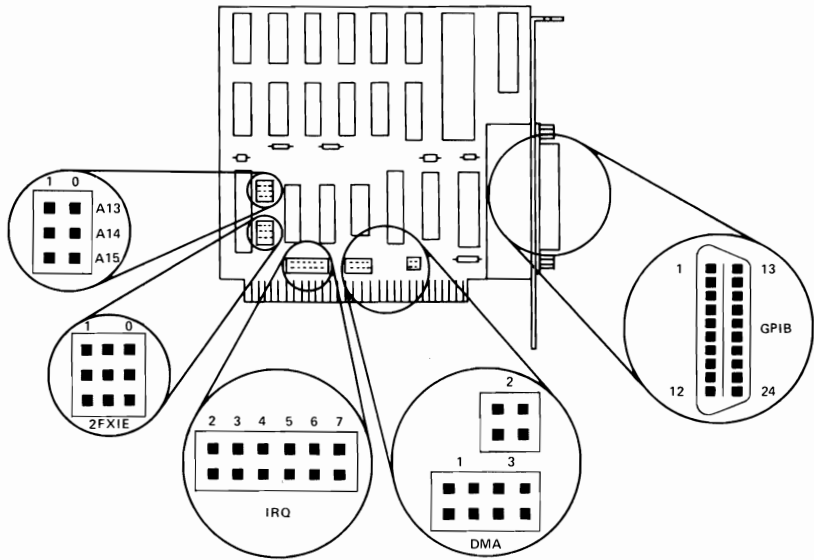
Programming information for the DMA Controller chip may be found in Intel's *Intel Component Data Catalog*. Additional information may also be found in the technical references for your system.

**Note:** At power-up time, the DMA Controller is initialized by the IBM PC BIOS program, which is in ROM on the system board. Programs written for processing GPIB Adapter DMA transfers must in no way change the overall configuration of the controller. Commands written to the controller should affect only the selected DMA channel.

The setup of the DMA Controller and the way it is used by the BIOS may be found in the BIOS program listing in your system's technical reference.

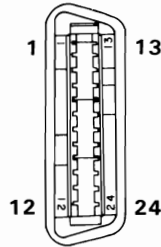
# Interface

The following illustration shows the GPIB Adapter with its connectors.



# Connector Specifications

The following table provides the pin numbers and their signals:



Signal Name/Description	Pin
DIO 1	1
DIO 2	2
DIO 3	3
DIO 4	4
EOI (24)	5
DAV	6
NRFD	7
NDAC	8
IFC	9
SRQ	10
ATN	11
SHIELD	12
DIO 5	13
DIO 6	14
DIO 7	15
DIO 8	16
REN (24)	17
Gnd, (6)	18
Gnd, (7)	19
Gnd, (8)	20
Gnd, (9)	21
Gnd, (10)	22
Gnd, (11)	23
Gnd, LOGIC	24

GPIB  
Device

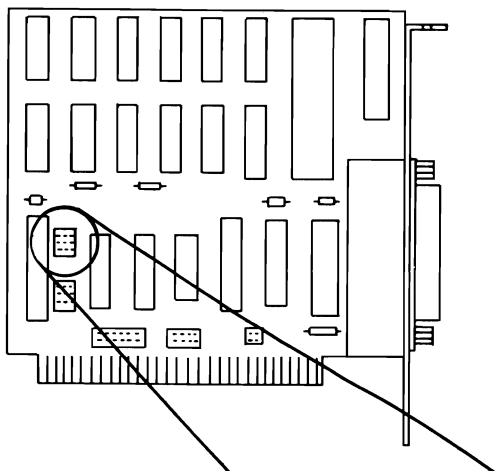
IBM  
GPIB  
Adapter

# Jumper Positions

This section describes the jumper positions that may be selected when installing the adapter.

# Adapter Selection

The following figure shows the jumper positions required for the adapter number of each adapter.

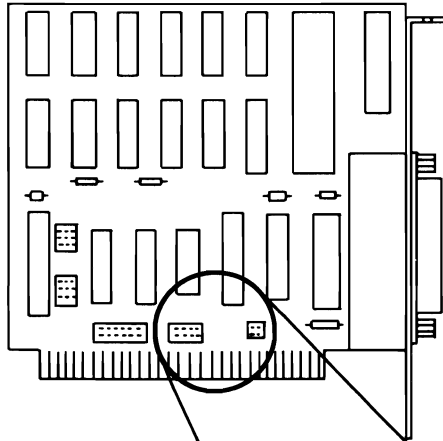



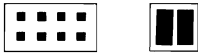

Adapter Number	Jumper Positions
0	
1	
2	
3	
4	
5	
6	
7	



# DMA Channel Selection

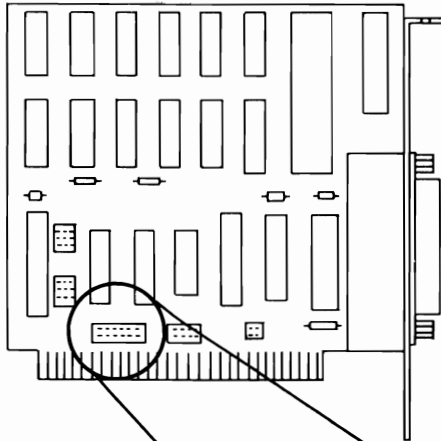
The following shows the jumper positions for selecting DMA channels.



Direct-Memory Access Channel	Jumper Positions
1	
2	
3	

# Interrupt-Request Selection

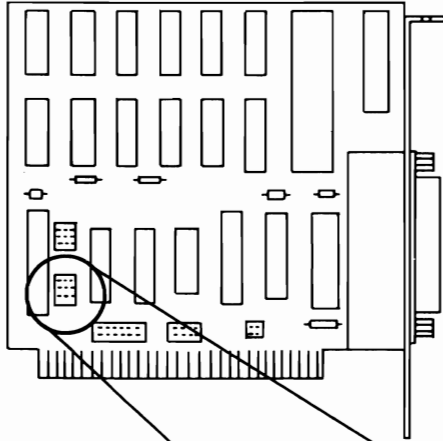
The following shows the jumper positions for selecting the interrupt-request line.









Interrupt-Request Level	Jumper Positions
7	
6	
5	
4	
3	
2	

# Interrupt-Acknowledge Selection

The following shows the jumper positions for each interrupt-acknowledge level.



Interrupt-Acknowledge Level	Jumper Positions
7	
6	
5	
4	
3	
2	

# Notes:



# Specifications

## Size:

Length: 11.43 cm (4.5 in)

Height: 10.67 cm (4.2 in)

Thickness: 1.57 cm (0.62 in)

Weight: 114.3 g (4.0 oz)

## Power Requirements:

Voltage: 5 Vdc (+/-5%)

Current: 750 mA Max, 400 mA Typical

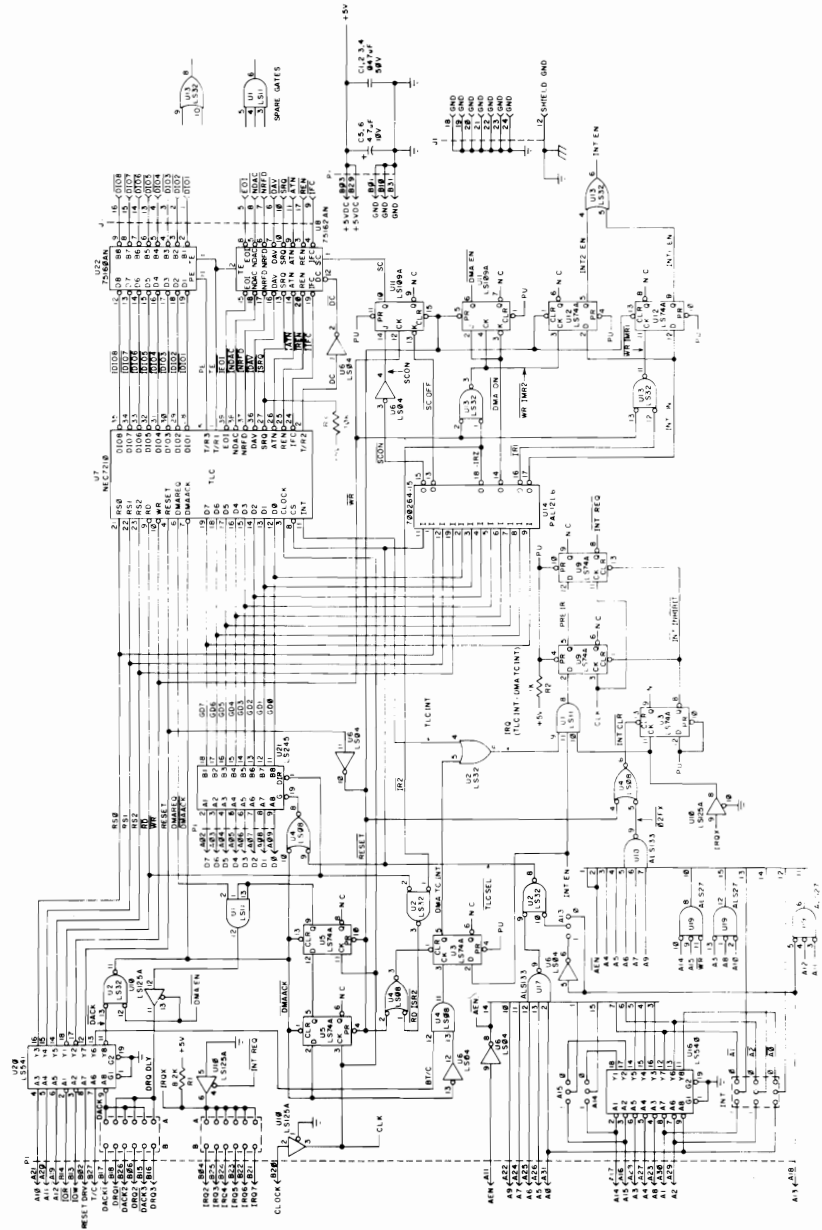
Power Dissipation: 3.75 Watts Max

# Notes:



# Logic Diagram

The following is the logic diagram of the GPIB Adapter.



# Notes:





# Index

## Special Characters

- μPD7210 (TLC) 13
  - read interface registers 13
  - write interface registers 19

## A

- address decode logic 12
- address mode register (ADMR) 21
- address mode 1 40
- address mode 2 41
  - programming the LE interface function 41
  - programming the TE interface function 41
- address mode 3 42
- address register (ADR) 29
- address register 0 (ADR0) 18
- address register 1 (ADR1) 18
- address selection 60
- address status register (ADSR) 16
- addressed implementation of talker and listener 39
  - address mode 1 40
  - address mode 2 41
  - address mode 3 42
- ADMR (address mode register) 21
- ADR (address register) 29
- ADR0 (address register 0) 18
- ADR1 (address register 1) 18
- ADSR (address status register) 16
- auxiliary mode register (AUXMR) 22
  - AUXRA (auxiliary register A) 26

AUXRB (auxiliary register B) 27  
AUXRE (auxiliary register E) 28  
ICR (internal counter register) 24  
PPR (parallel poll register) 25  
auxiliary register A (AUXRA) 26, 27  
auxiliary register E (AUXRE) 28  
AUXMR (auxiliary mode register) 22  
AUXRA (auxiliary register A) 26  
AUXRB (auxiliary register B) 27  
AUXRE (auxiliary register E) 28  
ICR (internal counter register) 24  
PPR (parallel poll register) 25  
AUXRA (auxiliary register A) 26, 27  
AUXRE (auxiliary register E) 28

## B

block diagram GPIB adapter 4

## C

CDOR (command/data out register) 19  
codes (interface codes supported) 1, 2, 3  
command pass through register (CPTR) 17  
command/data out register (CDOR) 19  
components  
     $\mu$ PD7210 TLC 13  
    address decode logic 12  
    control-signal buffer 5  
    data-bus buffer 5  
    DMA circuitry 6  
    GPIB transceivers 30  
    interface control logic 31  
    interrupt circuitry 7  
conducting a parallel poll 50  
conducting serial polls 45

configuring for a parallel poll 48  
control-signal buffer 5  
controller application 35  
CPTR (command pass through register) 17

## **D**

data transfer cases 36, 37  
data-bus buffer 5  
data-in register (DIR) 14  
DIR (data-in register) 14  
disabling parallel polling 50  
DMA channel selection 61  
DMA circuitry 6  
DMA controller 56  
DMA data transfer 43  
DMA T/C interrupt 11  
DMA transfers 55

## **E**

END or EOS (sending) 44  
END or EOS (stopping) 44  
end-of-string register (EOSR) 29  
EOSR (end-of-string register) 29

## **G**

GPIO transceivers 30

# I

- ICR (internal counter register) 24
- IEEE-488 interface function codes 1
- IMR1 (interrupt mask register 1) 19
- IMR2 (interrupt mask register 2) 20
- interface control logic 31
- interface information 57, 58
- interface registers 13
  - read only 13
    - ADR0 (address register 0) 18
    - ADR1 (address register 1) 18
    - ADSR (address status register) 16
    - CPTR (command pass through register) 17
    - DIR (data-in register) 14
    - ISR1 (interrupt status register 1) 14
    - ISR2 (interrupt status register 2) 15
    - SPSR (serial poll status register) 15
  - write only 19
    - ADMR (address mode register) 21
    - ADR (address register) 29
    - AUXMR (auxiliary mode register) 22
    - CDOR (command/data out register) 19
    - EOSR (end-of-string register) 29
    - IMR1 (interrupt mask register 1) 19
    - IMR2 (interrupt mask register 2) 20
    - SPMR (serial poll mode register) 20
- internal counter register (ICR) 24
- interrupt circuitry 7
  - DMA T/C interrupt 11
  - shared interrupt logic 8
- interrupt controller 54
- interrupt handler routine 53
- interrupt mask register 1 (IMR1) 19
- interrupt mask register 2 (IMR2) 20
- interrupt status register 1 (ISR1) 14
- interrupt status register 2 (ISR2) 15
- interrupt-acknowledge selection 63
- interrupt-request selection 62
- interrupts 52
- ISR1 (interrupt status register 1) 14
- ISR2 (interrupt status register 2) 15

## J

jumper positions

address selection 60

DMA channel selection 61

interrupt-acknowledge selection 63

interrupt-request selection 62

## L

logic diagram 67

## P

parallel poll register (PPR) 25

parallel polls 47

conducting 50

configuring 48

disabling 50

responding 51

physical characteristics 65

polling

parallel polls 47

serial polls 45

PPR (parallel poll register) 25

programmed I/O data transfer 43

programmed implementation of talker and listener 39

programmed initialization sequence 34

# R

- read registers 13
  - ADR0 (address register 0) 18
  - ADR1 (address register 1) 18
  - ADSR (address status register) 16
  - CPTR (command pass through register) 17
  - DIR (data-in register) 14
  - ISR1 (interrupt status register 1) 14
  - ISR2 (interrupt status register 2) 15
  - SPSR (serial poll status register) 15
- registers
  - read only 13
  - write only 19
- responding to a parallel poll 51
- responding to a serial poll 46

# S

- sending and receiving commands 43
  - DMA data transfer 43
  - programmed I/O data transfer 43
  - sending END or EOS 44
  - stopping on an END or EOS 44
- serial poll mode register (SPMR) 20
- serial poll status register (SPSR) 15
- serial polls 45
  - conducting serial polls 45
  - responding to serial polls 46
- shared interrupt logic 8
- shared interrupt logic timing diagram 10
- specifications 65
- SPMR (serial poll mode register) 20
- SPSR (serial poll status register) 15

## T

- talker or listener application 39
  - addressed implementation 39
  - programmed implementation 39
- talker/listener/controller [ $\mu$ PD7210-(TLC)] 13
  - read interface registers 13
  - write interface registers 19

## W

- write registers 19
  - ADMR (address mode register) 21
  - ADR (address register) 29
  - AUXMR (auxiliary mode register) 22
    - AUXRA (auxiliary register A) 26
    - AUXRB (auxiliary register B) 27
    - AUXRE (auxiliary register E) 28
  - ICR (internal counter register) 24
  - PPR (parallel poll register) 25
  - CDOR (command/data out register) 19
  - EOSR (end-of-string register) 29
  - IMR1 (interrupt mask register 1) 19
  - IMR2 (interrupt mask register 2) 20
  - SPMR (serial poll mode register) 20

**Notes:**

