W91284PIC

IEEE 1284 Peripheral Interface Controller

Data Sheet

Revision 4.00
Updated 29 October 1999

This is a data sheet for the W91284PIC and is subject to change. Warp Nine Engineering makes no warranties as to the correctness or fitness of this component for your application.

Warp Nine Engineering assumes no responsibility or liability for (1) any errors or inaccuracies contained in the information herein and (2) the use of the information or a portion thereof in any application, including any claim for (a) copyright or patent infringement or (b) direct, indirect, special or consequential damages. There are no warranties extended or granted by this document. The information herein is subject to change without notice from Warp Nine Engineering.

This document is property of Warp Nine Engineering and is provided for purposes of your review only. Please direct any questions to:

Larry Stein
Warp Nine Engineering
3645 Ruffin Road
Suite 330
San Diego, CA 92123

Ph. 619-292-2740
Ph. 619-292-2742 (direct)
Fax 619-292-8020

Web site: http://www.fapo.com
email: lstein@fapo.com

LIFE SUPPORT POLICY

WARP NINE ENGINEERING'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF WARP NINE ENGINEERING.

As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.

2. A critical component in any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

Revision History

Revision	Date	Notes
4.00	10/29/99	Clarification and editorial changes
3.31	5/12/99	Minor notes. No functional changes
3.30	5/7/99	Added Section 10, Errata, to deal with chip anomalies.
3.20	4/26/99	Added notes for Device ID.
3.10	2/25/99	Clarified DMA Direction, Pin types
3.01	1/28/99	Enhanced Dimension drawing
3.00	12/23/98	Added EPP_Read and EPP_Write Timeout interrupts to ISR2 Modified DMA Moved DC_Service_Request from ISR2[7] to IER2[7] Removed the Loop back function FDCR[7:6]
2.11	11/4/98	Updated DMA Write cycle timing diagram
2.10	11/3/98	Updated DMA Read cycle timing diagram
2.09	10/30/98	Added Byte mode discussion to Implementation notes Corrected Register read/write timing diagram
2.07	10/29/98	Many text changes. Added more Implementation Notes
2.01	10/12/98	Removed Ack Width from TCPR[5] Added nInit interrupt to ISR2
1.98		Internal
1.97	9/11/98	Added Implementation Notes
1.96	9/10/98	Added “DC Service Request” to ISR2 Changed default of ISR2 to 01h Changed the name of nLED to nSelected
1.95	9/3/98	Internal release
1.90	9/1/98	Cleaned up documentation
1.80	8/14/98	Deleted edge detect interrupts from ISR2 and IER2. Added Device_ID interrupt to ISR2/IER2 bit 0
1.70	8/9/98	Removed: Device_ID interrupt Restored: Data Receive interrupt
1.60	6/29/98	Clarified TC interrupt and added to Register Summary table Added: IRQ Assertion Level bit (PCCR:7) Daisy Chain Enable bit (PCCR:6)
1.50	6/16/98	Added: Terminal Count (TC) interrupt to ISR1 and IER1 Removed: Data Receive Interrupt from ISR1 and IER1
1.40	5/8/98	Baseline Added: notes regarding DMA usage

Table of Contents:

1.0 Overview

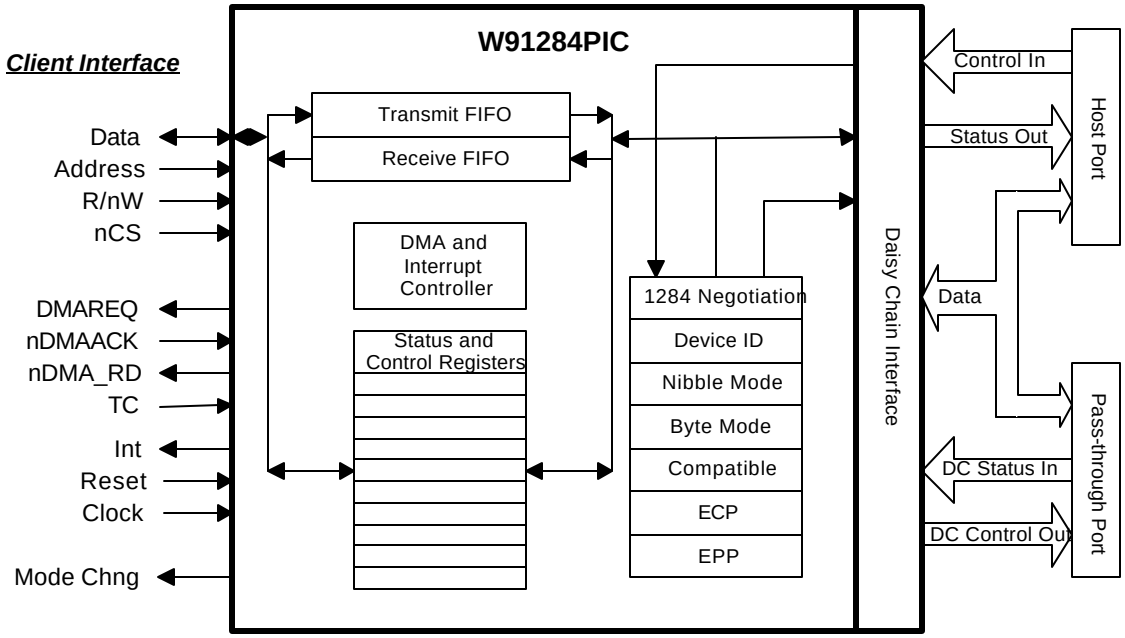


Figure 1 – W91284PIC Block Diagram

The W91284PIC is a an integrated solution that can be used to provide an IEEE Std. 1284-1994 interface for any parallel port peripheral. Specifically, the W91284PIC can be used with appropriate firmware to provide a fully compliant IEEE 1284 peripheral interface. This interface provides fast, bi-directional data transfer and control information when connected to a 1284 host parallel port. This IC is suitable for providing the peripheral interface for a wide variety of devices such as printers, scanners, multifunction and any other parallel port peripheral.

The functions of the W91284PIC implement the physical level hardware state machines and registers that interact with the host side of the interface and provide data transfer between the parallel port and the client side peripheral bus. The W91284PIC provides support for the following IEEE 1284 operational modes:

- IEEE 1284 Mode Negotiation
- Device ID
- Compatible Mode
- Nibble Mode
- Byte Mode
- EPP Mode
- ECP Mode (without RLE)

In addition to these IEEE 1284-1994 modes the W91284PIC provides support for a pass-through port that allows a printer or other peripheral to share the same host PC parallel port. This is provided via the IEEE 1284.3 port sharing standard. This is also known as Daisy Chaining. Figure 2 shows and

example of 4 daisy chain devices connected together sharing a single parallel port. At the end of the chain is a 'legacy' printer device.

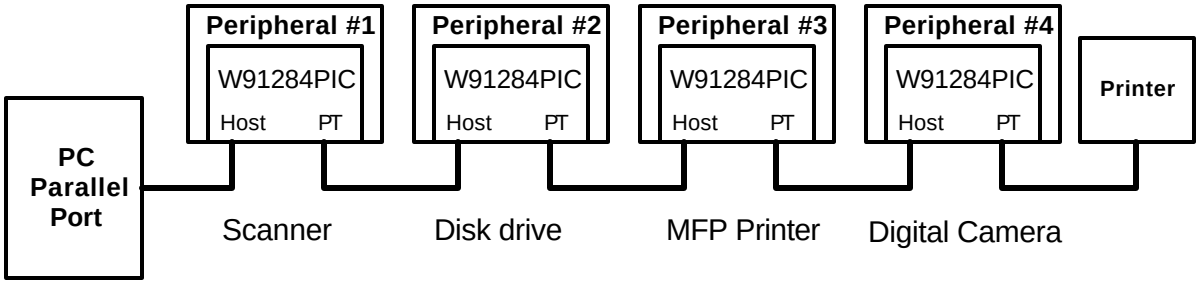


Figure 2 -- Daisy Chain Example

The W91284PIC implements the IEEE 1284 peripheral interface state machine in hardware. The state machine handles all of the 1284 signaling on the parallel port interface and acts as a data "bridge" to the client interface. The microcontroller firmware is not involved with the 1284 signaling.

There are a number of parameter registers that the firmware can set that will control the basic operation of the interface. These parameters include:

- Operational Modes available
- Device_ID available
- Compatible mode parameters
- Interrupt and DMA interfaces

The W91284PIC provides an IEEE 1284 Level II electrical interface, needing no external transceivers to interface to the cable. For peripherals requiring added protection there is support for external 74ACT1284 transceivers. As shown in figure 1, the W91284PIC interfaces to the 1284 parallel port on one side, and the client's peripheral interface on the other. From the client side, the W91284PIC looks like a 'generic' microcontroller interface. The basic operation of the interface is controlled via settings in the W91284PIC control registers. Once enabled, the W91284PIC provides all of the handshaking and data transfer to and from the parallel port. This data is transferred across client interface via DMA or PIO to the W91284PIC's transmit and receive FIFOs. These FIFOs are 16 bytes deep. It is not necessary for the client processor to have any knowledge of, or implement any of the IEEE 1284 parallel port protocols. This is all managed via the W91284PIC state machines.

The following sections will describe the functionality of the interfaces as well as the control registers. For more information on IEEE Std. 1284-1994 see reference 9.1.

1.1 Parallel Port Interface

The interface to the parallel port supports the standard I/O signals plus additional support for external transceiver control and the Daisy Chain interface. This interface may be connected to the parallel port of the host PC or to the pass-through port of a peripheral with daisy chain capability.

The W91284PIC may be used with external 74ACT1284 transceivers or drive the parallel port directly. A data direction pin is provided to interface to the external transceivers. The Host port connects to the parallel port of the PC or to the pass-through port of another daisy chain device. The pass-through

port of the W91284PIC provides an attachment interface for another parallel port peripheral. Up to four daisy chain devices may be attached to a single PC parallel port. Table 1 identifies the major signal groups.

Signal Name/Group	Direction	# of Pins
Parallel Port Data Lines	Bi-directional	8
Host Control Lines	Input	4
Host Status Lines	Output	5
Pass-through Control Lines	Output	4
Pass-through Status Lines	Input	5
External Direction	Output	1
External Output Enable	Output	1
Daisy Chain Function Disable	Input	1

Table 1 -- Peripheral Interface Pins

When the W91284PIC is used in a daisy chain environment the Host Control and Status lines are routed through the Host connector, and the Pass-through Control and Status lines are routed through the Pass-through connector. The parallel port data lines are shared by the two ports and make only one connection to the W91284PIC.

1.2 Client Interface

The W91284PIC is designed to be easily interfaced to any 8 bit microcontroller interface. This is a simple interface consisting of chip select, read/nWrite, and address. The W91284PIC has DMA capability that can be used to transfer blocks of data into and out of the module. Either DMA or Programmed I/O (PIO) may be used to transfer data. The chip supports status for CPU polling as well as interrupts.

The W91284PIC uses a simple register model and may be memory or I/O mapped. The chip select (nCS) is used to qualify the address lines and provide a data strobe. The R/nW line determines the direction of the transfer. Interrupts may be generated based upon the settings of the Interrupt Enable Registers, IER1 and IER2.

In this architecture, the 1284 peripheral interface is a simple register/memory model. The client firmware sets up the basic operational features and the state machine handles all of the data transfer and communication with the parallel port.

For example: The firmware sets up a mask that indicates the available operational modes. When a Negotiation start is detected the state machine will use the mask to compare to the requested mode. The state machine will either negotiate into the mode or indicate a failed negotiation. The system will not be alerted to the mode change until after it has occurred.

Table 2 shows the signals that are used for the client interface.

Signal Name/Group	Direction	# of Pins
System Data Lines	Bi-directional	8
Register Address Lines	Input	5
nCS - Chip Select	Input	1
R/nW	Input	1
DMA_REQ – DMA Request	Output	1
nDMA_ACK – DMA Acknowledge	Input	1
nDMA_RD – DMA Direction	Output	1
TC -- Terminal Count	Input	1
INT –Interrupt	Output	1
System Clock	Input	1
Mode_Chng -- Mode Change	Output	1
Imd_Term – Immediate Termination	Output	1
State Machine Reset	Input	1

Table 2 -- Client Interface Signals

2.0 W91284PIC State Machines

The W91284PIC is comprised of a number of separate state machines that control the various functions of the interface. Together they provide a complete 1284 to client interface. The various modules include:

- IEEE 1284 Peripheral State Machine
- IEEE 1284.3 Daisy Chain
- Client Interface

2.1 IEEE 1284 Peripheral Interface State Machine

This state machine manages the protocol on the parallel port interface. The state flow for this module is derived from the IEEE standard "*IEEE 1284-1994 Standard for a Bi-directional Peripheral Interface for Personal Computers*". This module implements the protocol requirements for a peripheral connected to a 1284 compliant interface. In addition, the device will function as a basic Centronics style interface for use with any older host adapter.

Figure 3 shows the general major 1284 phase transitions. The details of the transitions are not shown in this figure. For the actual state transition requirements please refer to the IEEE 1284-1994 standard.

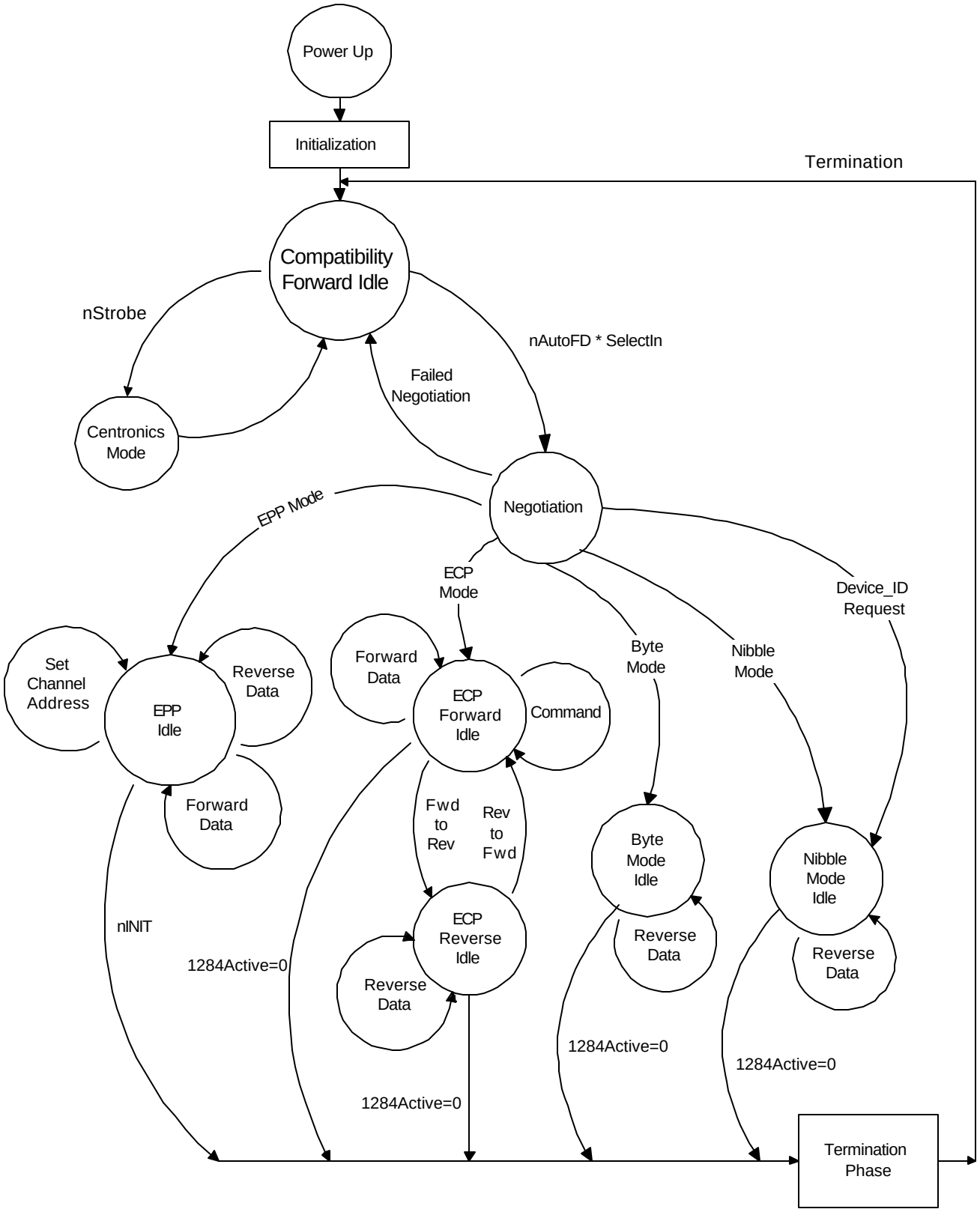


Figure 3 – 1284 Mode Transitions (simplified)

2.2 IEEE 1284.3 Daisy Chain

The W91284PIC provides the ability to share the parallel port with other peripherals. This chip provides an implementation of the daisy chain (DC) portion of the IEEE 1284.3 standard . See reference 8.2.

As shown in figure 2, daisy chaining is a port sharing method whereby a peripheral provides two ports (connectors). One port is for connection to the host port of the PC or to the pass-through port of another DC device. The other port provides the downstream connection for another DC device or a legacy parallel port peripheral.

Daisy chaining provides two major states for the peripheral. It is either in SELECTED or it is in TRANSPARENT mode. In the selected mode the DC state machine provides a direct connect from the peripherals' 1284 interface to the host port, while isolating downstream devices from the activity on the parallel port. It is in this mode that the peripheral driver communicates with the peripheral. When there is no need for direct communication then the interface is returned to the transparent (de-selected) mode. When in transparent mode, the host port of the peripheral is essentially connected to the pass-through port. This allows another device to use the parallel port.

The select, de-select, and addressing are communicated to the DC state machine by a protocol called the "Command Packet Protocol (CPP)." The CPP uses a sequence of self-qualifying data bytes, without additional control strobes, to communicate with the peripheral. Please refer to the IEEE 1284.3 standard for more details on this protocol.

The daisy chain feature of the W91284PIC can be disabled for peripherals that are not intended for port sharing. Please refer to section 4.6, *IEEE 1284.3 Daisy Chain Considerations*, for more information on this function.

2.3 Client Control Bus Interface

This state machine manages the interface to the client bus. This includes the DMA controller, interrupt monitor, and FIFO controller.

3.0 W91284PIC Register Model

The W91284PIC uses an address space of 32 contiguous registers. Currently only 17 of the registers are used. Unused registers are not implemented and should not be accessed. The registers are used to enable the W91284PIC operation and to control the functionality of the chip.

This section describes the registers and their usage. These registers can only be accessed by the client and are not visible by the 1284 host. Table 3 provides the register list.

Reg. #	Name	Description	R/W	Default
0	DATA	Data Port	R/W	N/D
1	CADR	Channel Address Register	R/W	N/D
2	PCCR	PIC Control Register	R/W	00
3	PCSR	PIC Status Register	R	00
4	NEMR	Negotiation Enable Mask Register	R/W	51
5	ISR1	Interrupt Source Register 1	R/W	00
6	IER1	Interrupt Enable Register 1	R/W	00
7	FDCR	FIFO Data Control Register	R/W	0C
8	FDSR	FIFO Data Status Register	R	05
9	SSCR	Centronics Status Signal Control Register	R/W	1A
A	CSSR	Centronics Control Signal Status Register	R	0x
B	TCPR	Timing Control Parameter Register	R/W	08
C	TFCR	Transmit FIFO Count Register	R	0x
D	RFCR	Receive FIFO Count Register	R	0x
E	ISR2	Interrupt Source Register 2	R/W	00
F	IER2	Interrupt Enable Register 2	R/W	00
10-1E		Undefined	N/D	XX
1F	CVR	Chip Version Register	R	20

Table 3 – Register Model

3.1 Data Port (DATA)

Base+00h: This register is read/write.

The data port is used to transfer data to and from the system bus to the PIC. The source and destination of the data is controlled by the FIFO Data Control register (FDCR).

If the FIFOs are enabled then access to this register will read and write to the receive and transmit FIFO respectively. If the FIFOs are disabled then access to this register will read or write the Data holding register. The data in this register is either input from the parallel port data lines or output to the parallel port data lines depending upon the state of the Manual_Data_Direction bit (FDCR[5]). When in the manual mode there are actually two registers, a data out register and a data in register. You cannot read back the data out register.

3.2 Channel Address Register (CADR)

Base+01h: This register is read/write.

This register is used for EPP and ECP mode only. Reading this register will read the data written by the host, not the client. There are actually two registers here, a Channel Address In and a Channel Address Out register. Therefore it is not a direct read/write register.

When a Address Command is received (forward direction) in either EPP mode (EPP Address Cycle) or in ECP mode (Command cycle with Address modifier) the data will be placed into this register, not the FIFO. This will cause an interrupt to be generated if enabled by the Interrupt Mask. **The interrupt will be asserted when DMA is idle or the interface changes to the reverse direction**.

In the Reverse direction, an EPP Address Read cycle will read this register. This register is used in conjunction with the PCSR bits 1 and 2. In ECP mode, a write operation will not cause any action to occur. In ECP the reverse channel address transfer is not supported in any of the current host chipset implementations.

In EPP mode a received address is in the range 0x00h to 0xFFh. In ECP mode, only Channel Addresses 0x00h through 0x7F are supported by the standard. In ECP, bit 7 is always high to indicate a Channel Address. Therefore, in ECP mode Channel Addresses 0x00h through 0x7F are read as 0x80 through 0xFF. The client should negate bit 7 when in ECP mode.

3.3 PIC Control Register (PCCR)

Base+02h: This register is read/write. Default to 00h

This register is used to control overall operation of the interface. A function is SET when the corresponding bit is set to a '1'.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Def	IRQ Level	DC Disable	Reverse Request	Auto Centronics En	DMA Dir	DMA En	Global Int En	Interface En

Bit 0: Interface Enable
This bit is set to enable the 1284 state machine to handle all interface transactions. This bit is cleared upon power up. The client sets this bit when any initialization is complete and the state machine can start.
This bit should only be changed when in Compatibility Mode.

Bit 1: Global Interrupt Enable
This bit is used to enable interrupts. This bit is used in conjunction with the Interrupt Mask registers.

Bit 2: DMA Enable
Set to enable DMA transfers. Used to enable DMA transfers to and from the FIFOs. If DMA is not enabled then the firmware may use interrupts or polling in order to determine data status and to transfer data.
DMA must be disabled before changing the DMA_Direction bit.

Bit 3: DMA Direction
This bit is used to control the direction of the DMA transfer.

Bit = 0: DMA forward enabled. From the receive FIFO to the client.
Bit = 1: DMA reverse enabled. From the client to the transmit FIFO.
This bit is output to the client interface as nDMA_RD.
This bit should only be changed when DMA Enable is disabled (bit 2=0).

Bit 4: Auto Centronics Enable
Set to enable the state machine to accept Centronics data while in the Compatible mode forward idle state. If not set then the client software is responsible for the interface handshaking.

Bit 5: Reverse Request
Setting this bit will cause the nFault line to be set when the host negotiates into ECP, EPP, nibble or byte mode. This indicates to the host that the client has data to send. This bit has no effect on the DMA transfers. This bit is also used to provide reverse data "packet boundary blocking" and to prevent the W92184PIC from entering the Reverse Idle Phase when in Nibble or Byte mode. Please refer to *4.4 Nibble and Byte Mode Reverse*.

Bit 6: DC Disable
Setting this bit will cause the Daisy Chain controller to be disabled and the interface to operate in the "always selected" mode. If the peripheral has a pass through port then this bit should be used for debug purposes only and never set during actual use. This bit is ignored if the external DC_nEN pin is set high, forcing the daisy chain function to be disabled.

Bit 7: IRQ Level
This bit determines the assertion level of the Interrupt pin.
Bit = 0: Interrupt is normally high and asserted low
Bit = 1: Interrupt is normally low and asserted high.

3.4 PIC Status Register (PCSR)

Base+03h: This register is read only. Defaults to 00h.

This register provides status information on the 1284 mode state and related status.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Def	Mode 3	Mode 2	Mode 1	Mode 0	Reverse	Addr In	Addr Out	1284 Active

Bit 0: 1284 Active
When set this bit indicates that this device interface is enabled, selected and ready. This bit is set when the Interface_Enable bit (PCCR[0]) is set and the daisy chain is in the Selected state.

Bit 1: Addr Out
This bit is set when the CADR is written to by the client. This bit is cleared when the host does an EPP Address read.

Bit 2: Addr In

This bit is set when an address byte is received from the host in either EPP or ECP mode. This bit is cleared when the CADR is read by the client controller.

Bit 3: Reverse

Set to '1' when the 1284 interface is in the Reverse (out) direction.

Bit [7:4] Interface mode

These bits are encoded and reflect the current operational phase or mode of the interface. Mode[3:0] are encoded as follows:

Mode_[3:0]	Current Mode
0	1284 State Machine disabled
1	Compatibility Mode – Deselected
2	Compatibility Mode – Selected
3	Negotiation to new mode
4	Terminating mode
5	Nibble Mode
6	Byte Mode
7	ECP (without RLE) Forward
8	ECP (without RLE) Reverse
9	Not Used
A	Not Used
B	EPP Mode
C	Device ID via Nibble Mode
D-F	Not Used

Table 4 – Mode Codes

3.5 Negotiation Enable Mask Register (NEMR)

Base+04h: This register is read/write. Default to 51h

The Negotiation Enable Mask is used by the state machine to determine which modes the interface will support. The W91284PIC will always be able to negotiate to Nibble mode. Setting the appropriate bit or bits in this register will allow negotiation into the mode(s). Some modes are not available (N/D)

Bit	7	6	5	4	3	2	1	0
Def	N/D	EPP	N/D	ECP w/o RLE	BECP (2)	Device ID(1)	N/D	Byte

(1) Device_ID, if supported, will only be supported in Nibble mode.

(2) BECP- Bounded ECP. A proposed mode of the IEEE P1284.3 standard. Not yet released. Not functional in this version of the W91284PIC. See 10.0 Errata

3.6 Interrupt Source Register 1 (ISR1)

Base+05h: This register is read/write. Default to 00h

This register is used to indicate the condition that generated the interrupt. A '1' indicates that the condition occurred. Reading this register will clear all interrupt indications and re-enable interrupts according to the IER1 register. The client should address all outstanding interrupt conditions when this register is read.

Bit	7	6	5	4	3	2	1	0
Def	Xmit FIFO Empty	Recv FIFO Full	Rev2 Fwd	DMA TC	RLE Error	Address Recv	Data Receive	Mode Change

Bit 0: Mode Change Interrupt
If this bit is set then an interrupt was generated do to a mode change. The Interface Mode Status register reflects the current mode. The client should check this register to determine if the mode change requires a Device ID string response.

Bit 1: Data Receive Interrupt
If this bit is set then an interrupt will be generated whenever a data or command byte is received from the host.

Bit 2: Address Received
If this bit is set then an interrupt was generated when an ECP Command cycle with address modifier or an EPP Address Write occurred.

Bit 3: RLE Error
If this bit is set then an interrupt was generated due to the 1284 interface receiving a RLE Count byte although this mode is not supported by the chip.

Bit 4: DMA TC
If this bit is set then an interrupt was generated due to a DMA Terminal Count being indicated on the TC input.

Bit 5: Rev2Fwd
This bit is set when the 1284 interface is in the reverse direction with DMA enabled and a mode change occurs that causes the 1284 transfers to then be in the forward direction.

Bit 6: Receive FIFO Full
If this bit is set then an interrupt was generated when the Receive FIFO goes full.

Bit 7: Transmit FIFO Empty
If this bit is set then an interrupt was generated when the Transmit FIFO goes empty.

3.7 Interrupt Enable Register 1 (IER1)

Base+06h: This register is read/write. Default to 00h

This register is used to enable a interrupt to be generated on the condition defined by the Interrupt Register 1. Writing a 1 to the appropriate bit will enable that condition.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Def	Xmit FIFO Empty	Recv FIFO Full	Rev2 Fwd	DMA TC	RLE Error	Address Recv	Data Receive	Mode Change

3.8 FIFO Data Control Register (FDCR)

Base+07h: This register is read/write. Default to 0Ch

The W91284PIC includes a Transmit FIFO and a Receive FIFO. The FIFOs are 16 bytes deep. The FIFO Data Control register is used to control the FIFO and to direct the destination for any parallel port data received by the module. The available targets for the data are the receive FIFO, or discard. Discard may be used for interface performance measurements or for testing.

If the FIFOs are not enabled then the data must be read or written from the data port on a per byte basis. The FIFOs must be enabled for normal operation. The client should always enable and disable the FIFOs together and treat it as a single enable.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Def	N/D	N/D	N/D	Manual Data Dir	Xmit FIFO Enable	Recv FIFO Enable	Xmit FIFO Reset	Recv FIFO Reset

Bit 0: Receive_FIFO Reset.
Set to 1 to clear the FIFO and the Receive FIFO counter. Must be reset to 0 to enable the FIFO.

Bit 1: Transmit_FIFO Reset.
Set to 1 clear the FIFO and the Transmit FIFO counter. Must be reset to 0 to enable the FIFO.

Bit 2: Receive_FIFO Enable.
When set the Receive_FIFO is enabled.

Bit 3: Transmit_FIFO Enable.
When set the Transmit_FIFO is enabled.

Bit 4: Manual Data Direction
If the Transmit and Receive FIFOs are disabled then this bit controls the direction of the DATA Port (register 0). A ‘0’ sets the direction to forward (in) and a read of the DATA port will read the data on the parallel connector data lines. A ‘1’ sets the direction to reverse (out) and the parallel connector data lines will reflect

the last value written to the DATA port. This should only be used when the 1284 state machine is disabled and the client is driving the protocol (PCCR[0] = 0).

Bit 5: Not defined
Bit 6: Not defined
Bit 7: Not defined

3.9 FIFO Data Status Register (FDSR)

Base+08h: This register is read only. Default to 05h.

This register provides status of the Receive and Transmit FIFOs.

Bit	7	6	5	4	3	2	1	0
Def	N/D	N/D	N/D	N/D	Xmit FIFO Full	Xmit FIFO Empty	Recv FIFO Full	Recv FIFO Empty

Bit 0: Receive FIFO empty.
Bit 1: Receive_FIFO full.
Bit 2: Transmit FIFO empty.
Bit 3: Transmit FIFO Full.
Bit 4: Not Used
Bit 5: Not Used
Bit 6: Not Used
Bit 7: Not Used

3.10 Centronics Status Signal Control Register (SSCR)

Base+09h: This register is read/write. Default to 1Ah

These bits are defined for Compatibility Mode and EPP Mode only. Bits [4:0] are valid only when the 1284 state machine is disabled and the client is managing the parallel port data transfer. Bits [7:5] are valid when the mode is Compatible and the 1284 state machine is enabled. Bit 0 is valid in EPP mode as a User Defined signal.

These bits are non-inverting and will set the output signal to the level indicated.

Bit	7	6	5	4	3	2	1	0
Def	Error 2	Error 1	Pause Busy	Ack	Busy	PE	Fault	Select

Bit 0: Select
This bit sets the Select out line to 1.
User defined in EPP mode.
Bit 1: Fault
Setting this bit will assert the nFault line to 1.
Bit 2: PE (Paper Empty or Paper Error)

This bit sets the PE out line to 1.

Bit 3: Busy
This bit sets the Busy out line to 1.

Bit 4: Ack
Setting this bit to 1 will assert the nAck line to 1.

Bit 5: Pause_Busy
Setting this bit will cause the PIC to assert the Busy line when operating in the Compatible Centronics mode.

Bit 6: Error Condition 1
Setting this bit will cause the PIC to asset Busy, PE and Fault when operating in the Compatible Centronics mode.

Bit 7: Error Condition 2
Setting this bit will cause the PIC to asset Busy, and Fault when operating in the Compatible Centronics mode.

3.11 Centronics Control Signal Status Register (CSSR)

Base+0Ah: This register is read only.

This register reflects the state of these signals at the connector interface. These bits are not logically adjusted and reflect the signal level at the interface.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Def	0	0	0	0	nSelectIn	nAutoFD	nInit	nStrobe

Bit 0: nStrobe
Bit 1: nInit
Bit 2: nAutoFD
Bit 3: nSelectIn
Bit 4: 0
Bit 5: 0
Bit 6: 0
Bit 7: 0

3.12 Timing Control Parameter Register (TCPR)

Base+0Bh: This register is read/write. Default to 0Ah

The W91284PIC is designed to operate at a maximum input clock frequency of 40MHz. The state machine may operate over any input clock rate between 16MHz and 40MHz. The performance may degrade with system clocks less than 16MHz.

This register provides a scalar for different system clocks and controls the function of the interface while in Centronics FIFO mode.

Bit	7	6	5	4	3	2	1	0
Def	Ack Pos 1	Ack Pos 0	N/D	CLKW4	CLKW3	CLKW2	CLKW1	CLKW0

Bit 0: CLKW0
Bit 1: CLKW1
Bit 2: CLKW2
Bit 3: CLKW3
Bit 4: CLKW4

These bits are used to adjust the timing parameters for different system clock speeds. The value of this field indicates the number of clock periods necessary to make a 500nS window.
For example: at 20MHz it takes 10 clocks for 500nS so the value of CLKW[4:0] should be set to 0Ah (10 decimal).

Bit 5: Not Used
Bit 6: Ack_Pos_0
Bit 7: Ack_Pos_1

These bits determine the relationship between the nAck pulse and the Busy status line. The nAck width is 500nS. See the following table and figure for this relationship.. See figure 4.

Ack_Pos_1	Ack_Pos_0	Position
0	0	No nAck pulse generated
0	1	nAck before Busy
1	0	nAck during Busy
1	1	nAck after Busy

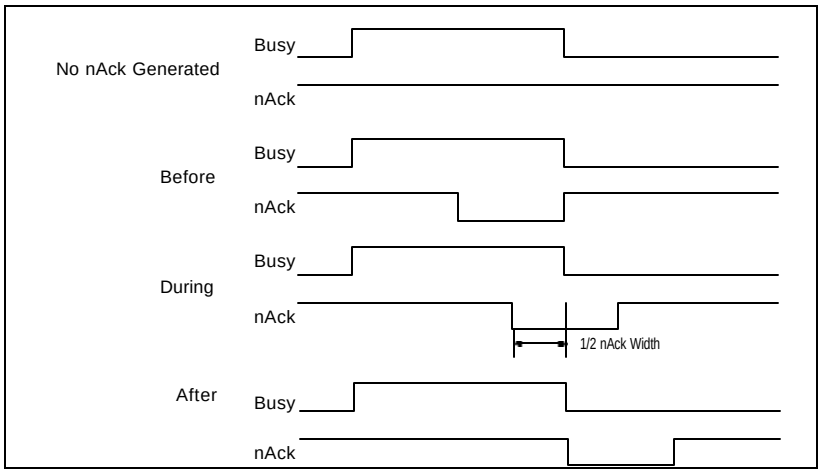


Figure 4 – nAck Position

3.13 Transmit FIFO Count Register (TFCR)

Base+0Ch: This register is read only.

This is an 8 bit register that contains the number of valid bytes in the Transmit FIFO. A value of '0' indicates that there are no bytes in the FIFO. A value of 1 through 10h indicates the number of bytes remaining in the FIFO.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Def	N/D	N/D	N/D	XCnt 4	XCnt 3	XCnt 2	XCnt 1	XCnt 0

Bit 0: XCnt 0
Bit 1: XCnt 1
Bit 2: XCnt 2
Bit 3: XCnt 3
Bit 4: XCnt 4
Bit 5: 0
Bit 6: 0
Bit 7: 0

3.14 Receive FIFO Count Register (RFCR)

Base+0Dh: This register is read only.

This is an 8 bit register that contains the valid number of bytes in the Receive FIFO. A value of '0' indicates that there are no bytes in the FIFO. A value of 1 through 10h indicates the number of bytes remaining in the FIFO.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Def	N/D	N/D	N/D	RCnt 3	RCnt 3	RCnt 2	RCnt 1	RCnt 0

Bit 0: RCnt 0
Bit 1: RCnt 1
Bit 2: RCnt 2
Bit 3: RCnt 3
Bit 4: RCnt 4
Bit 5: 0
Bit 6: 0
Bit 7: 0

3.15 Interrupt Source Register 2 (ISR2)

Base+0Eh: This register is read/write. Default to 00h

This register is used to indicate the condition that generated the interrupt. A '1' indicated that the condition occurred. Reading this register will clear all interrupt indications and re-enable interrupts according to the IER2 register. The client should address all outstanding interrupt conditions when this register is read.

Bit	7	6	5	4	3	2	1	0
Def	N/D	N/D	N/D	N/D	EPP Write Timeout	EPP Read Timeout	nInit Asserted	Device ID Int

Bit 0: Device ID
If enabled, set to indicate that the current Mode_Change interrupt is due to a Device ID request from the host.

Bit 1: nInit Asserted
If enabled, set to indicate that nInit has been asserted while in Compatibility Mode. May be used to determine if there has been a system reset.

Bit 2: EPP_Read_Timeout
If enabled, set to indicate that the W91284PIC detected a host timeout while in the EPP_Read phase.

Bit 3: EPP_Write_Timeout
If enabled, set to indicate that the W91284PIC detected a host timeout while in the EPP_Write phase.

Bit 4: Not used

Bit 5: Not used

Bit 6: Not used

Bit 7: Not used

3.16 Interrupt Enable Register 2 (IER2)

Base+0Fh: This register is read/write. Default to 00h

This register is used to enable a interrupt to be generated on the condition defined by the Interrupt Register 2. Writing a 1 to the appropriate bit will enable that condition.

Bit	7	6	5	4	3	2	1	0
Def	DC Service Request	N/D	N/D	N/D	EPP Write Timeout	EPP Read Timeout	nInit Asserted	Device ID Int

Bit 0: Device ID

Bit 1: nInit

Bit 2: EPP_Read_Timeout

Bit 3: EPP_Write_Timeout
Bit 4: Not used
Bit 5: Not used
Bit 6: Not used
Bit 7: DC Service Request
Setting this bit will cause the Daisy Chain interface to indicate an interrupt condition when queried. This bit will be reset to 0 when the service request is sent and/or the Daisy Chain driver issues a Clear_Interrupt_Latch command. The client software may clear this request by setting this bit to 0.

3.17 Undefined

Base+10h through 1Eh:

These registers are unused. If accessed it will return indeterminate data.

3.18 PIC Version Register (PCVR)

Base+1Fh: This register is read only.

This register contains a revision control number for the W91284PIC. The value contains a major and minor revision level.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Def	0	0	1	0	0	0	0	0

Bit [3:0]: Minor [3:0]
Bit [7:4]: Major [3:0]

3.19 Register Summary

Offset	Name	Default	7	6	5	4	3	2	1	0
00	**DATA**	**N/A**	Parallel Data Port							
01	**CADR**	**N/A**	Command/Address Port							
02	**PCCR**	**00**	IRQ Assert Level	Daisy Chain Disable	Reverse Request	Auto Cent En	DMA Dir.	DMA En	Global Int En	Interface En
03	**PCSR**	**00**	Mode 3	Mode 2	Mode 1	Mode 0	Reverse	Addr In	Addr Out	1284 Active
04	**NEMR**	**51**	N/D	EPP	N/D	ECP w/o RLE	BECP	Device ID	N/D	Byte
05	**ISR1**	**00**	Xmit FIFO Empty	Recv FIFO Full	Reverse to Fwd	DMA TC	RLE Error	Address Recv	Data Receive	Mode Change
06	**IER1**	**00**	Xmit FIFO Empty	Recv FIFO Full	Reverse to Fwd	DMA TC	RLE Error	Address Recv	Data Receive	Mode Change
07	**FDCR**	**0C**	N/D	N/D	N/D	Manual Data Dir	Xmit FIFO Ena	Recv FIFO Ena	Xmit FIFO Reset	Recv FIFO Reset
08	**FDSR**	**05**	N/D	N/D	N/D	N/D	Xmit FIFO Full	Xmit FIFO Empty	Recv FIFO Full	Recv FIFO Empty
09	**SSCR**	**1A**	Error 2	Error 1	Pause Busy	nAck	Busy	PE	nFault	Select
0A	**CSSR**	**0x**	N/D	N/D	N/D	N/D	nSelectIn	nAutoFD	nInit	nStrobe
0B	**TCPR**	**08**	Ack Pos 1	Ack Pos 0	N/D	CLKW4	CLKW3	CLKW2	CLKW1	CLKW0
0C	**TFCR**	**0x**	N/D	N/D	N/D	XCnt 4	XCnt 3	XCnt 2	XCnt 1	XCnt 0
0D	**RFCR**	**0x**	N/D	N/D	N/D	RCnt 4	RCnt 3	RCnt 2	RCnt 1	RCnt 0
0E	**ISR2**	**00**	N/D	N/D	N/D	N/D	EPP Write Timeout	EPP Read Timeout	nInit Asserted	Device ID Int
0F	**IER2**	**00**	DC Service Request	N/D	N/D	N/D	EPP Write Timeout	EPP Read Timeout	nInit Asserted	Device ID Int
10-1E		Not Defined								
1F	**PCVR**	**20**	Major 3	Major 2	Major 1	Major 0	Minor 3	Minor 2	Minor 1	Minor 0

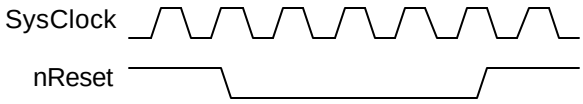
Table 5 – W91284PIC Register Summary

4.0 Implementation Notes

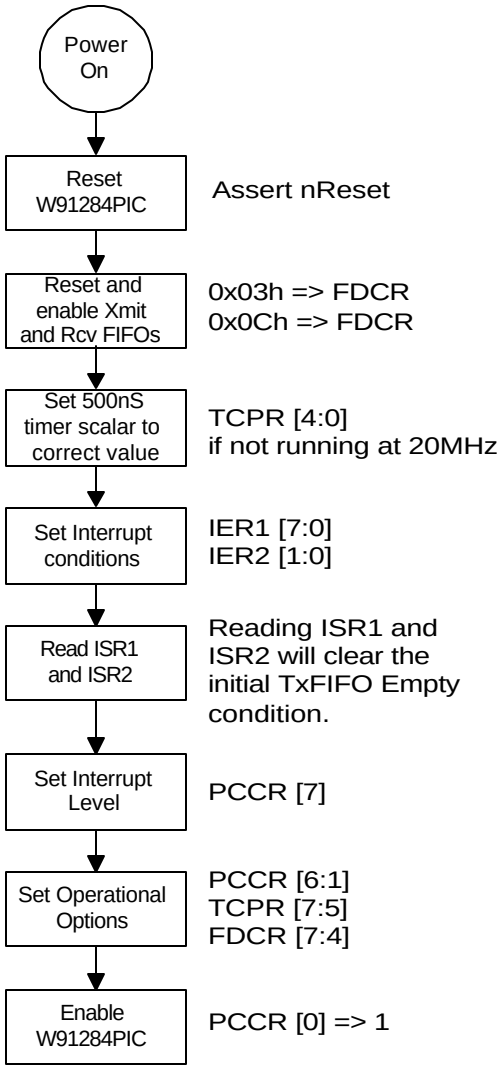
This section will provides information on implementation and use of the W91284PIC in a system environment.

4.1 W91284PIC Setup and Initialization

The initial setup for theW91284PIC is fairly straight forward. Upon power up, the client interface should assert nReset (pin 51) for a minimum of 4 rising clock edges, prior to accessing any of the PIC control registers.



Once the client has reset the chip, the process is to initialize the control registers and then enable the interface controller. The following flow diagram is an example of how the client may initialize the controller.



At this point the W91284PIC is operational and will respond to IEEE 1284 protocols on the parallel port. Prior to enabling the PIC the client software should set up any DMA or IRQ processes as necessary. If Auto_Centronics_En (PCCR [4]) is set, then the interface will respond to the standard Centronics protocol when in the Compatible mode, forward idle state.

4.2 DMA Interface

The W91284PIC provides a DMA capability for data transfer between the client and the transmit or receive FIFOs. The DMA operation is controlled by the DMA_Enable bit (PCCR[2]) and the DMA_Direction bit (PCCR[3]). In a 1284 environment, the current direction of the data across the cable is determined by the host PC. The implication of this is that the client (peripheral) cannot control which direction data is flowing, and therefore needs to respond to changes in data flow.

If DMA is enabled, then it is only enabled for either the forward or reverse direction, not for both. How the DMA controller responds is dependent upon the state of the interface. The following case examples will explain this operation.

Case #1: Interface in forward direction with DMA Forward enabled (PCCR[3]=0).
In this situation the W91284PIC will accept host data unless the receive FIFO is full, and will request a DMA transfer whenever the receive FIFO is not empty. If the host then negotiates to the reverse direction then the DMA request cycles will continue until the receive FIFO is empty. The client will be notified via the Mode_Change interrupt that a mode change has occurred. The client should then disable DMA and change the direction of the DMA transfer.

Case #2: Interface in forward direction with DMA Reverse enabled (PCCR[3]=1).
In this situation the W91284PIC will accept host data unless the receive FIFO is full but will not initiate any Forward DMA transfers. The client will need to use programmed I/O to read data from the receive FIFO It is the responsibility of the client software to be sure to empty the receive FIFO if necessary. Reverse DMA request cycles will be issued until the transmit FIFO is full.

Case #3: Interface in reverse direction with DMA Reverse enabled.
The W91284PIC will issue DMA request cycles whenever the transmit FIFO is not full. Reverse data will be transferred to the host PC. The host may terminate the reverse transfer phase at any time. If the host terminates back to forward mode then no more data will be sent from the transmit FIFO to the host, but DMA request cycles will be issued until the transmit FIFO is full or until DMA is terminated by the client.

Case #4: Interface in reverse direction with DMA Forward enabled.
In this case the client software must use programmed I/O to keep data in the transmit FIFO. No DMA will be active. If the host terminates back to forward mode then no more data will be sent from the transmit FIFO to the host, but DMA may be requested for the forward direction. See case #1.

The DMA interface consists of a DMA request line (DMA_REQ) , DMA acknowledge (nDMA_ACK), DMA direction output (nDMA_RD) and a Terminal Count line (TC). DMA is enabled by the client by setting the DMA_Enable bit (bit 2) and the DMA_Direction bit (bit 3) in the PIC Control Register.

The DMA channel is assigned by the client DMA controller. Therefore nDMA_ACK may only be asserted for this channel. The DMA_Direction bit is output as the nDMA_RD signal and is used to indicate the direction for the DMA request. Chip select is not necessary for a DMA cycle to proceed and **must be held high** during DMA acknowledge cycles. TC is used to signal the end of the DMA transfer block from the client to the W91284PIC. The Mode_Change signal may be used to signal an end to a forward DMA due to a mode change as the result of a termination phase. Please note that the host may end the forward data transfer but remain in the idle state and not change modes or cause a termination phase to occur. The client will have to determine if the forward transfer is complete.

4.3 EPP Mode Operation

One of the areas where EPP differs from ECP operation is how reverse channel data is transmitted from the peripheral to the host PC. In ECP operation, the host negotiates to ECP forward channel and then does a negotiation to ECP reverse. See figure 3. Once in ECP reverse, the peripheral SENDS data, if available, to the host PC. The host may not "know" if there is data to be received or how much data there is. The nFault line is asserted if the W91284PIC has data in the transmit FIFO or if the Reverse Request bit is set. This is used to indicate to the host that the peripheral has data to send. The interface stays in reverse mode until the host determines that all the data has been received or terminates the reverse data transfer and re-enters the ECP forward idle phase. In this way, the host can receive unsolicited reverse data of indeterminate length. ECP mode was designed for peripherals, such as printers, that operate in this manner.

In EPP operation, the host negotiates to the EPP idle phase. From this phase the host can initiate either reads or writes of data or address. Rather than the peripheral sending data, as in ECP, the host READS data from the peripheral. In EPP, the host needs to know how much data there is to read before it starts to read. In EPP mode there is a critical response time of 10uS in which the peripheral must respond to an access or an EPP timeout may occur on the host PC. This is due to how EPP cycles are generated on the PC ISA bus. When an EPP cycle starts the ISA bus is held in wait states until the peripheral responds. If the peripheral does not respond within the 10uS limit the cycle will complete without a valid data transfer occurring. This is to keep the ISA bus from being held up and causing memory problems on the PC.

So, unlike ECP, EPP cycles are closely coupled to the device driver. This puts a requirement on the host driver to always "know" how much data to read, and when it will be ready. In the IEEE 1284-1994 standard there is no nDataAvailable (nFault) signal defined for use in this manner with EPP. The standard provides three User Defined signals:

- PError
- nFault
- Select

The W91284PIC uses two of these signals to enhance the reverse channel capability of the EPP mode.

4.3.1 EPP Mode nFault

For EPP mode the W91284PIC uses the nFault line as a nDataAvailable signal. This signal will be asserted whenever the interface is in EPP mode and either one or both of the following conditions are met:

1- There is data in the Transmit FIFO (Tx_FIFO not empty)
2- The Reverse_Request bit is set (PCCR [5] = 1)

This allows the host driver to determine if there is reverse data available before attempting any EPP data read cycles. This does not affect EPP Address read cycles. EPP Address reads are from the CADR and will always complete.

If the client software (PIC side) uses the Reverse_Request bit, then it is suggested that the client put data into the Tx_FIFO prior to setting the Reverse_Request bit. This is to ensure that the W91284PIC will respond to an EPP Data read without delay. If this bit is used, then the client must be able to ensure that a Tx_FIFO Empty condition will not exist for more than 8uS while Reverse_Request is asserted.

4.3.2 EPP Mode PError

For EPP mode the W91284PIC uses the PE as a "Transmit FIFO full" indication to the host. This provides the host PC with the ability to read a burst of 16 bytes when PE and nFault are both asserted.

4.3.3 Host Driver Considerations for EPP Mode

Given the enhancements that the W91284PIC includes for EPP mode, there are some things that the host driver may take into consideration. The following notes are from the Warp Nine Engineering 1284 Toolkit driver product (version 2.70 and greater), but may be implemented by any host driver.

1- EPP data or address writes (forward direction):

Nothing for the host to do except write the data. The W91284PIC will throttle the transfer rate depending upon the state of the Receive FIFO. This could result in an EPP TIMEOUT in the forward direction, but there is nothing the host driver can do about that. It is up to the client application to be sure that the peripheral can take the data and read data from the receive FIFO. A Receive_FIFO_Full condition should not be allowed for longer than 8uS.

If a timeout occurs this will generate an EPP_Write_Timeout interrupt, if enabled. Once an EPP timeout occurs the host may still transfer data but this data may not be valid. The host software must reset the timeout error on the host controller in order reset this condition.

2- EPP Address read (reverse direction):

Address reads should be done one at a time. This is a low bandwidth channel. The W91284PIC only supports 1 address cycle at a time and does not use the Tx_FIFO to send address information. The client driver is responsible for maintaining "fresh" data in the CADR.

3- EPP Data read (reverse direction):

This is where the enhancement makes the biggest difference. Now there are three conditions that the host can consider for EPP:

A- Ignore nFault
B- Use nFault and not PE
C- Use nFault and PE

A- Ignore nFault

This condition is the same as IEEE 1284 EPP implementation. The PC driver will read however many bytes the PC application requested. If the W91284PIC cannot keep up with the data rate then an EPP TIMEOUT may occur. Otherwise it will complete normally.

If a timeout occurs this will generate an EPP_Read_Timeout interrupt, if enabled. Once an EPP timeout occurs the host may still request data. If the transmit FIFO is not empty then the W91284PIC will transfer valid data to the host.

B- Use nFault but not PE

This can be used when the host driver knows how much data to read and that the peripheral can keep up. In this case the driver will negotiate to EPP and wait for nFault to be asserted. Once asserted then the driver will read the number of bytes requested. This could still result in a EPP TIMEOUT if the peripheral stalls. This would occur if the PIC client software does not respond to a Tx_FIFO Empty condition within 8uS.

C- Use nFault and PE

This is the best case. In this mode the peripheral may have any size blocks to send and the driver will be able to determine when there is data available and how much. The PC client driver will have to indicate to the host driver how large the peripheral transmit FIFO is (16 in the case of the W91284PIC) and the value of a background "block transfer timeout."

The driver should wait for nFault and PE to be asserted and then read 1 FIFO block size. The driver will reset the timer on each block read and then poll on PE and nFault and then repeat. While polling, if the block timer triggers and nFault is asserted but not PE then the driver should read one byte at a time until nFault goes high.

Following these guidelines will result in a more robust EPP interface with a reduced chance of interface timeout conditions with maximum possible performance.

4.4 Nibble and Byte Mode Considerations

4.4.1 Nibble and Byte Mode Reverse Idle

This section pertains to the use of Nibble and Byte mode for reverse channel data. If the system is using ECP or EPP then reverse data indication is available while in the idle state of those modes.

When the IEEE 1284 interface is in the Compatible idle state there is no indication from the peripheral as to whether or not it has reverse data available. Typically the host will periodically negotiate to either Nibble or Byte mode to determine if there is data available. The IEEE 1284 protocol uses the

nFault line indicate the presence of reverse data. If nFault is asserted (low) then there is data available, if high, then there is no data available. At this point there are two options depending upon the state of nFault:

- If nFault is asserted:
  - Transfer data from peripheral to host
- If nFault is not asserted:
  - Terminate back to Compatible mode and try later
  - Wait in the Reverse_Idle_Data_Not_Available phase.

When in the Reverse_Idle_Data_Not_Available phase the host usually will set a background timer which will limit the time that host will wait for data to become available. If the timer expires then host will terminate back to Compatible. During this time peripheral data may become available. The peripheral indicates this by going through the "Reverse_Idle_Interrupt_Phase". This phase generates an interrupt to the host by pulsing the nAck signal low for 500nS. After the Interrupt_Phase handshake the interface enters the reverse data transfer phase and the peripheral data is sent to the host. Figure 5 shows the relationship between these phases.

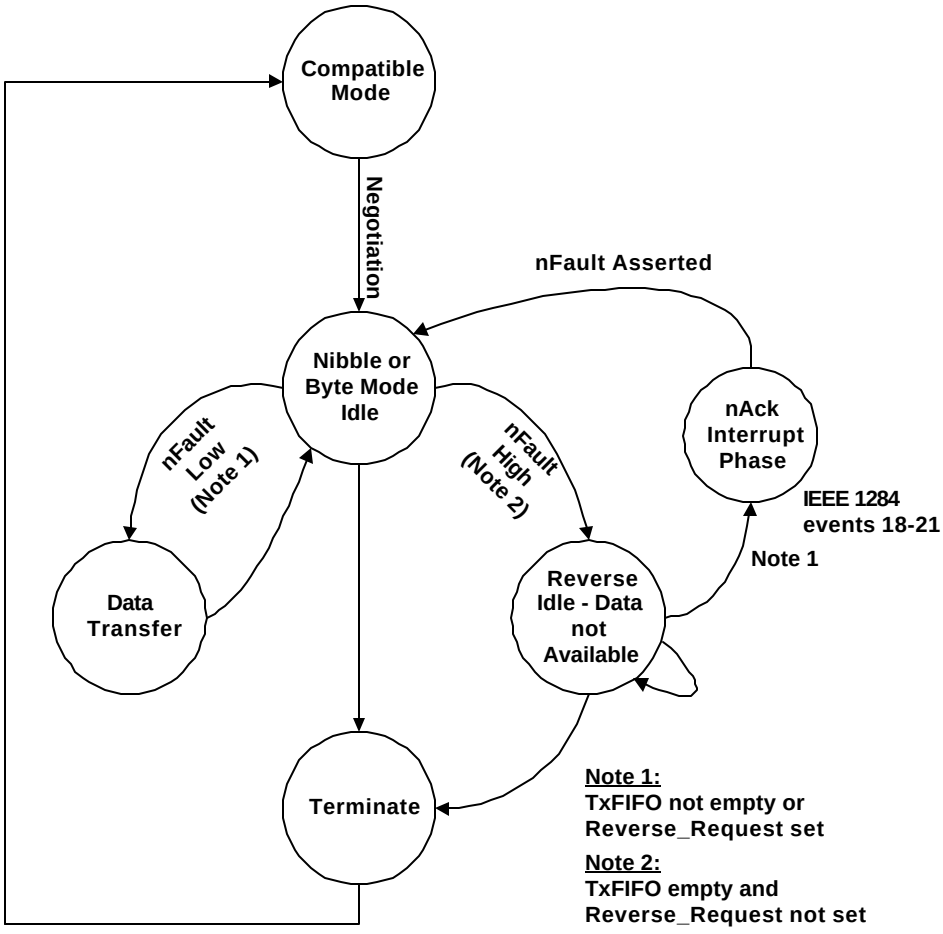


Figure 5 – Reverse Idle Phase Transitions

There are two items that control the state of the nFault signal while in any of the reverse modes. These are the transmit FIFO empty status and the Reverse_Request bit (PCCR[5]). If the Reverse_Request bit is not set then the nFault signal will follow the state of the TxFIFO_E signal (FDSR[2]). The effect of this is that in Nibble or Byte modes if the transmit FIFO goes empty at the end of a byte transfer, then the W91284PIC will enter the Reverse_Idle_Data_Not_Available phase. To exit this phase the interface will have to go through the Interrupt Phase. This may have performance implications.

If the peripheral is sending a large or known block size of reverse data then it is preferable to not have the interface going through the Interrupt Phase. If the TxFIFO goes empty for just the time it takes the client to respond to a TxFIFO_E interrupt then the Interrupt phase adds unnecessary overhead to the reverse transfer.

With the Reverse_Request bit set the nFault signal will be asserted independent of the transmit FIFO status. This means that the interface will stay in the Data Available phase throughout the entire reverse data transfer. The only requirement is that the client respond to the TxFIFO_E interrupt or write data into the transmit FIFO within 35mS of going empty. The Reverse_Request bit can be used to "block" or "bound" a given amount of reverse data. Another way of viewing this is to think of it as a way to indicate a block of data for a particular job.

See Section 10 Errata.

4.4.2 Byte Mode Deviation from the IEEE 1284 Standard

The IEEE 1284 Byte mode provides a method to transfer reverse data using the parallel ports' data lines. This mode was made possible by the introduction of a data direction control bit on the IBM PS/2 host parallel port. Byte mode is capable of providing the same reverse transfer rate as the Centronics mode does in forward.

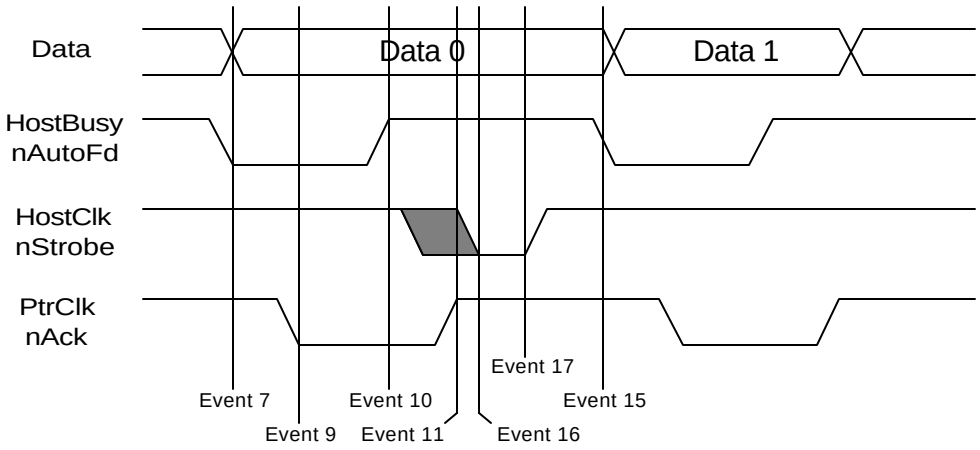


Figure 6 – Byte Mode Modification

Figure 6 shows a simplified view of the IEEE 1284 defined Byte mode data transfer and the Byte mode as implemented in the W91284PIC. The data transfer starts with the host setting HostBusy low to indicate that it can accept data (event 7). The peripheral responds by putting data on the data bus, waiting 500nS, and then setting PtrClk low to indicate that valid data is on the data bus (event 9). The host reads the data and sets HostBusy high to indicate that it has read the data (event 10). The peripheral sets PtrClk

high to acknowledge that the host has read the data (event 11). The host then, or coincident with event 10, sets HostClk low (event16) and then high (event 17). At that point the handshake is complete.

Note that events 16 and 17 occur without any corresponding handshake from the peripheral. These events were defined to make the Byte mode handshake compatible with some IBM PS/2 parallel port DMA modes. Current ECP Printer Ports and drivers do not implement this nStrobe pulse.

In order to make the handshake more efficient and maintain better data transfer rates, the W91284PIC does not need the nStrobe pulse. The data is considered accepted at event10 and the state machine will change the data after event 11. For drivers that may implement events 16 and 17 there is no adverse affect since the data is accepted at event10.

4.5 Device ID

The W91284PIC supports negotiation for IEEE Device ID. The device ID is used by Windows 95 and 98 operating systems to enable auto-detection for devices attached to the parallel port. While the IEEE 1284 standard enables support for the device ID to be returned in either Nibble, Byte or ECP mode, the W91284PIC only supports Nibble mode for returning the device ID string. The client indicates support for Device_ID by setting the Device_ID bit of the Negotiation Mask Register, NEMR[2].

When the W91284PIC receives a negotiation request for Device_ID_Nibble_Mode the chip will indicate a Mode Change interrupt with the Device_ID request interrupt asserted and enter the Nibble mode phase. If there is no data in the transmit FIFO, the interface will enter the Reverse_Idle_Data_Not_Available state. In this case, when the client software writes the Device_ID string data into the Transmit FIFO the interface will go through the Interrupt phase. This sequence is used to indicate to the host that reverse data is now available. If upon entering the Nibble mode there is data in the Transmit FIFO then this data will be sent to the host. This means that non-Device_ID data can be sent as the Device ID.

In general, the Device_ID request would happen upon system initialization. In this instance it is not very likely that the wrong information will be sent. One method to resolve this potential problem is to not pre-load the Transmit FIFO. The client may set the Reverse_Request bit (PCCR[5]) to indicate that reverse data is available but not write the data into the Transmit FIFO until the interrupt is received and decoded. The performance impact of this is minimal. See section 4.4.1 Nibble and Byte Mode Considerations.

The Device ID string is fully defined in IEEE Std.1284-1994. The device ID is an ASCIIZ string that peripherals may send to the host as a result of a successful negotiation. The ID is intended to enable the automatic detection of peripheral devices, such as printer type, and allow the automatic selection of the appropriate device driver.

The ID string is made up of a length field and then a sequence of ASCIIZ strings with the format:

key: value {,value};

Each key will have at least one value. The minimum keys are **MANUFACTURER**, **COMMAND SET** and **MODEL**. These keys are case sensitive and may be abbreviated as **MFG**, **CMD** and **MDL**. Additional keys may be added at the discretion of the manufacturer.

The length bytes are 2 hex bytes that identify the length of the string including the length bytes. The first byte is the most significant, and the second byte the least significant Following is an example of a device ID:

007d
MANUFACTURER:Warp Nine Engineering;
COMMAND SET:none;
MODEL:F/Mux;
DRIVER:fmux.sys;
COMMENT:For tech support call 619/292-2740;

It is not necessary that a peripheral support the device ID in order to be compliant with the 1284 specification, but it is highly recommended that all new designs implement this capability.

4.6 IEEE 1284.3 Daisy Chain Considerations

This section is provided as a brief description of how the IEEE 1284.3 interface operates and to provide any W91284PIC explicit information. For complete information on the IEEE 1284.3 standard please refer to reference 9.2.

The W91284PIC has full support for the Daisy Chain (DC) portion of the IEEE 1284.3 standard. As shown in figure 2, Daisy Chaining is a method for sharing the parallel port with other peripherals. The 1284.3 protocol used to control the DC interface is called the Command Packet Protocol (CPP).

The CPP packet consists of a sequence of data bytes presented on the parallel port data lines without any intervening transitions on the nStrobe line or any other host control line. The 1284.3 compliant peripheral responds by asserting various parallel port status lines during the sequence. When the host recognizes the correct peripheral response then it is able to issue a DC command and assert the nStrobe to clock the command into the peripherals' DC controller. The last DC device on a chain of 1284.3 devices will block this nStrobe from going out onto its' pass through port and being interpreted as data by an attached printer. The CPP data sequence looks like:

AA	**55**	**00**	**FF**	**87**	**78**	**Command**	**FF**

The command byte in the CPP packet represents a command code and possibly an address as well. ***aa*** refers to DC device address 0 – 3. The defined codes are shown in table 6. All DC devices shall implement the mandatory commands. If implemented, all optional commands must be implemented as a group. The W91284PIC supports all of these commands.

DC CPP command codes	CPP command codes	Operation	Implemented
(0x00-0x03)	0000 00aa	Assign Address aa to the current device	Mandatory
(0xE0-0xE3)	1110 00aa	Select Device aa	Mandatory
(0x30)	0011 0000	De-select device	Mandatory
(0x08-0x0B)	0000 10aa	Query Interrupt from device aa	Mandatory
(0x48)	0100 1000	Enable Daisy Chain Interrupts	Optional
(0x40)	0100 0000	Disable Daisy Chain Interrupts	Optional
(0x58-0x5B)	0101 10aa	Set Interrupt Latch on device aa	Optional
(0x50-0x53)	0101 00aa	Clear Interrupt Latches on device aa	Optional

Table 6 -- Daisy Chain CPP command codes

Action	DC Device 1 State	DC Device 2 State	Host - Std 1284 Mode
Power Up	Un-Addressed Un-Selected	Un-Addressed Un-Selected	Compatibility Mode
Assign Address	Addressed 0 Un-Selected	Addressed 1 Un-Selected	Compatibility Mode
Select Device 1	Selected	Un-Selected	Compatibility Mode
Negotiate	Selected	Un-Selected	New Mode
Data Transfer	Selected	Un-Selected	New Mode
Terminate	Selected	Un-Selected	Compatibility Mode
De-Select Device	Addressed Un-Selected	Addressed Un-Selected	Compatibility Mode
Select Device 2	Un-Selected	Selected	Compatibility Mode
Negotiate	Un-Selected	Selected	New Mode
Data Transfer	Un-Selected	Selected	New Mode
Terminate	Un-Selected	Selected	Compatibility Mode
De-Select Device	Addressed Un-Selected	Addressed Un-Selected	Compatibility Mode

Table 7 -- Daisy Chain device state table

DC devices default to transparent mode until addressed and selected. Upon selection, the peripheral and host port are in the Compatibility Mode. Once selected, the peripheral may be negotiated into another mode. The interface shall always terminate to Compatibility Mode prior to de-selection. See table 7 for an example of DC device state changes. In this example, DC device 1 is attached to the host and DC device 2 is attached to the pass-through port of DC device 1. There may be legacy device attached to the pass-through port of DC device 2.

Once a device is addressed, that address remains in effect until a new Assign Address command is completed. In the un-selected state the DC device is considered transparent and shall pass through and not alter Data, Control or Status lines. In selected state, the selected DC peripheral shall be the only device actively communicating with the parallel port host. The state of the output control lines is latched at the time of selection. While selected, the pass through port of the DC device shall maintain the value of the output control lines in this latched state.

When selected, the host port status lines of the DC device reflect the state of the DC device, not that of the pass through port. When not selected, the host status lines reflect the status of the DC device pass-through port.

When a CPP preamble has been detected the host status lines reflect the appropriate status response for the DC CPP protocol.

Figure 7 shows the general states for a complete communication cycle for a 1284.3 device. In this architecture there is a clear separation between the responsibilities of the Daisy Chain driver and the IEEE 1284 driver.

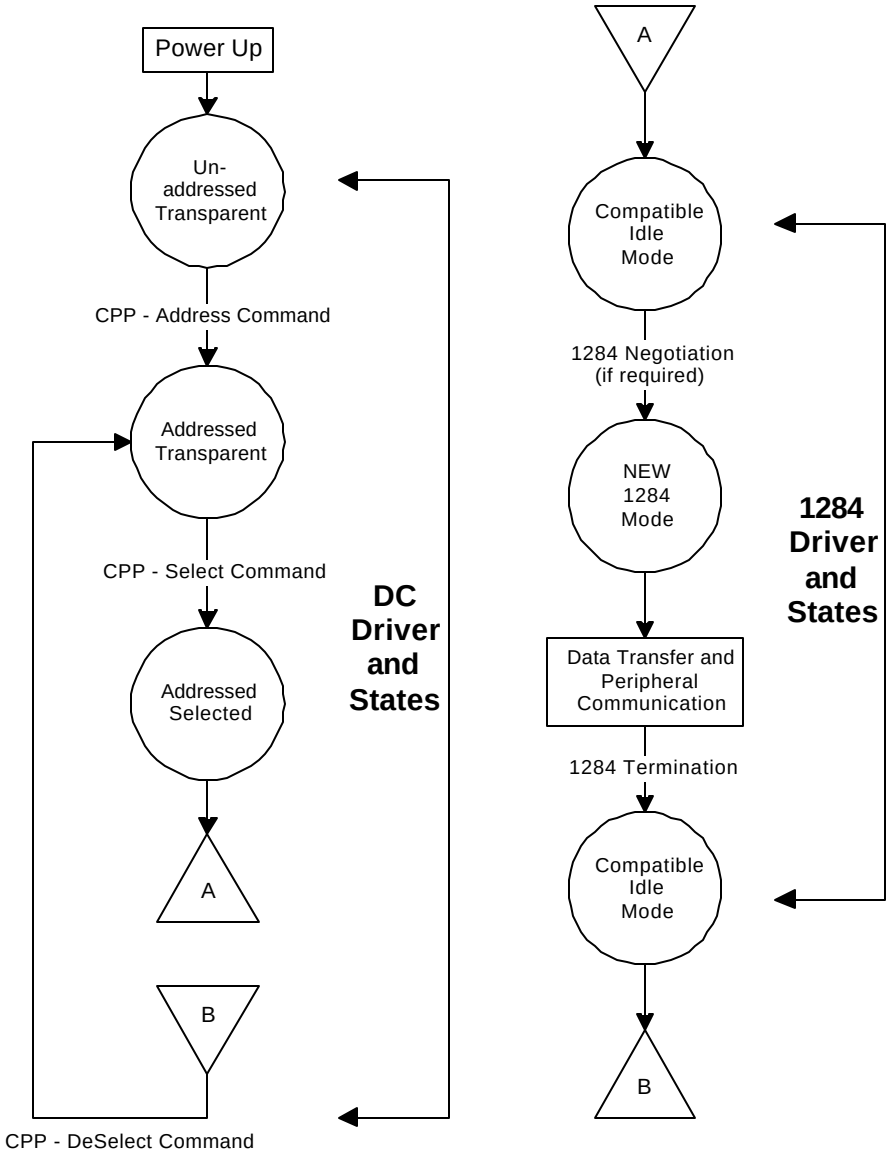


Figure 7 – Daisy Chain Device Communication States

The client can determine if the peripheral is Selected by checking the 1284 Active bit (PCSR[0]). If set, then this device is selected.

4.6.1 Daisy Chain Global Reset

When a host is reset, it may reinitialize the interface by asserting nInit low in conjunction with nSelectIn low for a minimum of 10 uS. The W91284PIC will recognize this condition and if selected, will become de-selected. If assigned, the DC address will remain and the interface will respond to a DC Select. A peripheral using the W91284PIC should not use the combination of nSelectIn low and nInit low.

The client can determine this condition by using the nInit interrupt (ISR2[1]) and the mode status, PCSR[7:4].

4.7 Parallel Port and Daisy Chain Interrupts

When a peripheral device is sharing the parallel port with other devices it is not always possible to generate an interrupt to the host. The parallel port is being used as a shared resource and the peripheral architecture should take this into consideration.

The IEEE 1284.3 architecture does not guarantee real time interrupts. A DC device may only generate an interrupt under the following conditions:

- The DC device is selected: In this case it may generate an interrupt as defined for its current IEEE 1284 mode.
- The DC device is not selected: In this case it shall only generate a DC interrupt during the time bounded by the reception of the Enable Interrupts command and the Disable Interrupts command.

DC interrupts are treated as Service Request indications. The CPP command "Query Interrupts" is used to poll the daisy chain and determine which devices have outstanding interrupts. Figure 8 shows how interrupts are enabled and disabled for a typical sequence of selecting and de-selecting a DC device. Upon power up, interrupts from the device are disabled and remain in this state until after DC addresses are assigned. The Enable DC Interrupts command allows interrupts to occur. The Enable DC interrupts command shall only be issued when no DC devices are selected. If DC interrupts are enabled, a Disable DC Interrupts command shall be issued before a device may be selected.

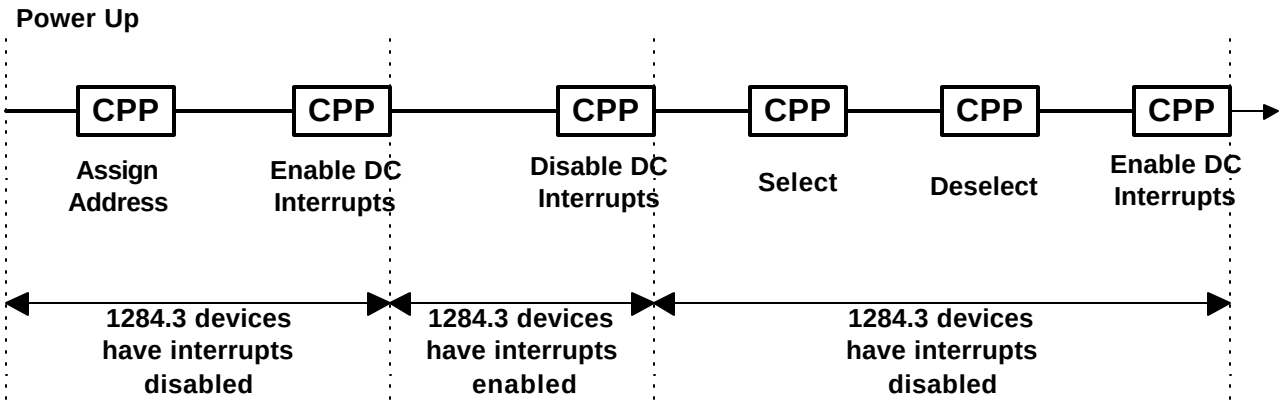


Figure 8 -- Interrupt Commands

A client can enable a DC interrupt by setting the DC Service Request bit, ISR2[7]. If this bit is set then the DC controller will issue an interrupt when the Enable DC Interrupts command is received. The host will respond by issuing the Query Interrupts command and then the Clear Interrupts command. Upon receipt of the Clear Interrupt Latch command the DC Service Request bit will be reset.

It is the responsibility of the host Daisy Chain software to read the interrupt latches of the DC devices and issue an interrupt to each of the required drivers.

5.0 W91284PIC Pin Definitions

Figure 9 shows the I/O pins used to interface to the W91284PIC.

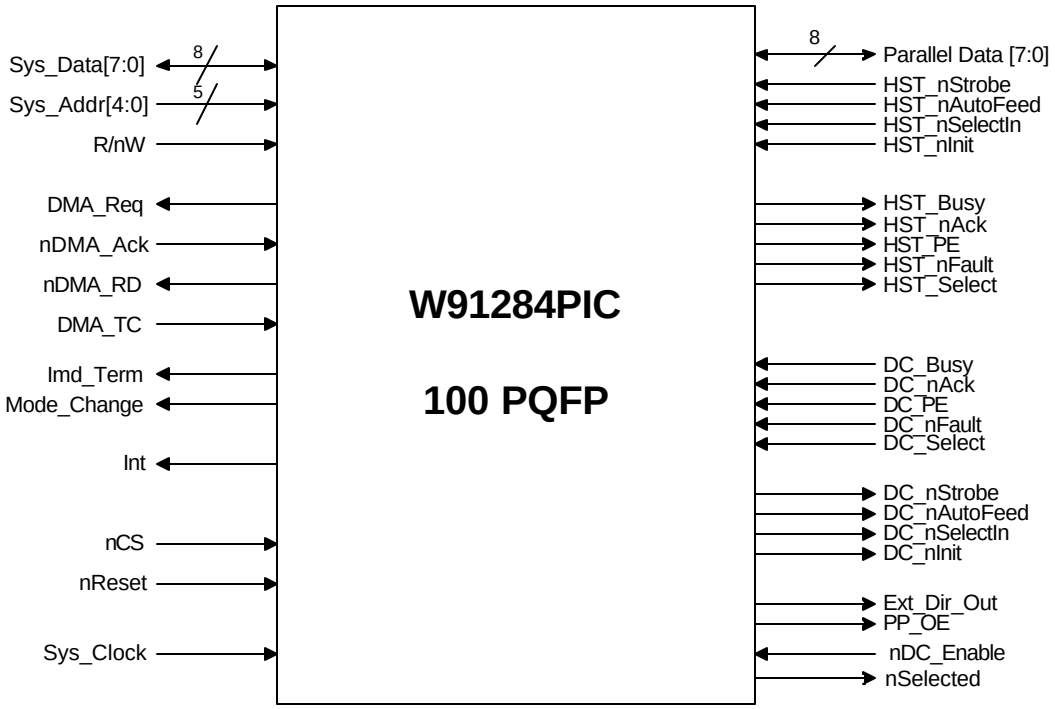


Figure 9 – I/O Signals

Table 8 lists the pin numbers and definitions for the W91284PIC. Unused pins are not listed and should not be connected to any signal.

I/O Pin Type Definitions:

I	Input pin	TTL level input
I_{SCH}	Input pin	TTL level input with Schmitt Trigger
O8	Output pin	8mA sink/source
O16	Output pin	16mA sink/source
$O16_{SR}$	Output pin	16mA sink/source IEEE 1284 Level II drive with slew rate limited outputs 0.05-0.40V/ns
B8	Bi-directional pin	Bi-directional I/O pin. 8mA sink/source
B16	Bi-directional pin	16mA sink/source IEEE 1284 Level II drive with slew rate limited outputs 0.05-0.40V/ns

The signals are **described as used in the Compatible/Centronics mode** of the IEEE Std. 1284-1994 standard. Please refer to that standard for a detailed description of the signal usage as it applies to various modes of operation (Reference 9.1).

Pin Definitions:

Pin #	Name	Type	Description
1	n/c		Not Connected - Do not connect to unused pins. All non-defined pins should be treated as no connects.
2	HST_Select	$O16_{SR}$	Select signal to host PC. Used in response to HST_nSelectIn to signal that the device is selected and on-line
3	HST_PE	$O16_{SR}$	PE signal to host PC. Used with HST_nFault to indicate an error condition
4	HST_Busy	$O16_{SR}$	Busy signal to host PC. Asserted to indicate that the peripheral cannot accept any data at this time.
5	HST_nAck	$O16_{SR}$	nAck signal to host PC. Asserted after a data strobe to indicate that the data has been accepted (optional). Also used to indicate that the peripheral needs attention.
6	HST_nSelectIn	I_{SCH}	Host signal to peripheral. Asserted to select the currently connected device.
7	HST_nInit	I_{SCH}	Host signal to peripheral. Asserted to indicate to the peripheral that it should enter the initialization or reset condition.
8	HST_nFault	$O16_{SR}$	nFault signal to host PC. May be used with HST_PE and HST_Busy to indicate a peripheral error condition.
9	HST_nAF	I_{SCH}	Host signal to peripheral. Used to indicate to printer that it should insert a line feed after a carriage return. Not used for that anymore.
10	HST_nStrobe	I_{SCH}	Host signal to peripheral. Asserted to indicate that valid data is on the parallel port data lines. Used with HST_Busy to send data in the Centronics mode.
11	PP_Data_0	B16	Bi-directional parallel port data bit 0
12	PP_Data_1	B16	Bi-directional parallel port data bit 1
13	PP_Data_2	B16	Bi-directional parallel port data bit 2
14	GND		
15	Vcc		
16	PP_Data_3	B16	Bi-directional parallel port data bit 3
17	PP_Data_4	B16	Bi-directional parallel port data bit 4
18	PP_Data_5	B16	Bi-directional parallel port data bit 5
19	PP_Data_6	B16	Bi-directional parallel port data bit 6
20	PP_Data_7	B16	Bi-directional parallel port data bit 7
21	DC_nStrobe	$O16_{SR}$	Pass-through daisy chain nStrobe
22	DC_nAF	$O16_{SR}$	Pass-through daisy chain nAF
23	DC_nFault	I_{SCH}	Pass-through daisy chain device signal nFault
24	DC_nInit	$O16_{SR}$	Pass-through daisy chain nInit
25	DC_nSelectIn	$O16_{SR}$	Pass-through daisy chain nSelectIn
26	DC_nAck	I_{SCH}	Pass-through daisy chain device signal nAck

27	DC_Busy	I_{SCH}	Pass-through daisy chain device signal Busy
28	DC_PE	I_{SCH}	Pass-through daisy chain device signal PE
29	DC_Select	I_{SCH}	Pass-through daisy chain device signal Select
30			
31	GND		
32	nPP_OE	O8	Parallel Port Data Output Enable (low). Can be used to drive an external transceiver. Note: This line is always asserted low and does not need to be connected. Use Ext_Dir_Out to control external transceivers.
33	Ext_Dir_Out	O8	External Direction Out – When high, the data channel is in the reverse direction, peripheral to host (out). When low, the data channel is in the forward direction, host to peripheral (in).
34	nSelected	O16	Asserted low when the device is Selected in daisy chain mode. This can be used to drive a LED.
35			
36	Do not use		Reserved
37			
38			
39			
40	Vcc		
41	GND		
42			
43			
44			
45			
46	Vcc		
47			
48	nDC_Enable	I	Tied high to disable the daisy chain state machine and place the peripheral in the always selected state. Pulled low to enable daisy chaining.
49	N/C		Do not use
50	Sys_Clk	I	W91284PIC system clock. A 16MHz to 40MHz CMOS or TTL oscillator may be connected to this pin.
51	nReset	I	Asserted low to reset the chip to it's initialization state. **Must be asserted after power up.**
52	DMA_REQ	O8	DMA Request – Asserted high to indicate that W91284PIC requires a DMA cycle. For a DMA read operation the DMA_REQ line is de-asserted when valid data is on the Sys_Data bus in response to the assertion of the nDMA_ACK signal. For a DMA write operation this signal is de-asserted when the chip has accepted the data on the Sys_Data bus.
53	nDMA_ACK	I	DMA Acknowledge – Asserted in response to the assertion of the

			DMA_Req line. For a DMA read operation this signal indicates that the W91284PIC may place valid data on the Sys_Data bus. This signal is de-asserted when the system has read the data. For a DMA write operation the assertion of this signal indicates that the system has placed valid data on the Sys_Data bus.
54	nDMA_RD	O8	DMA Read -- Asserted low to indicate that the requested DMA cycle is for a forward transfer, DMA read. When high, the requested DMA cycle is for the reverse transfer, DMA write.
55	DMA_TC	I	DMA Terminal Count – **OPTIONAL** - Asserted by the client to indicate the end of a DMA block transfer. This signal is used to generate an interrupt to the client. If not used then should be held low.
56	GND		
57	Vcc		
58			
59			
60	nCS	I	Chip Select – Used with programmed I/O to access the W91284PIC registers and the the FIFOs. Indicates that a valid address is on the Sys_Addr lines. On a write cycle nCS indicates that valid data is on Sys_Data bus. Data is latched on the falling edge of nCS for a write cycle. On a read cycle this signal indicates that the W91284PIC may place valid data on the Sys_Data bus. **For DMA cycles, nCS must be held high.**
61	R/nW	I	Read/notWrite – Valid when nCS is asserted. Indicates a read cycle when high, and a write cycle when low. Not used with DMA.
62	Sys_Addr_0	I	System Address bit 0
63	Sys_Addr_1	I	System Address bit 1
64	Sys_Addr_2	I	System Address bit 2
65	Vcc		
66	GND		
67	Sys_Addr_3	I	System Address bit 3
68	Sys_Addr_4	I	System Address bit 4
69	Sys_Data_0	B8	Bi-directional system data bus bit 0
70	Sys_Data_1	B8	Bi-directional system data bus bit 1
71	Sys_Data_2	B8	Bi-directional system data bus bit 2
72	Sys_Data_3	B8	Bi-directional system data bus bit 3
73	Sys_Data_4	B8	Bi-directional system data bus bit 4
74	Sys_Data_5	B8	Bi-directional system data bus bit 5
75	GND		
76	Sys_Data_6	B8	Bi-directional system data bus bit 6
77	Sys_Data_7	B8	Bi-directional system data bus bit 7

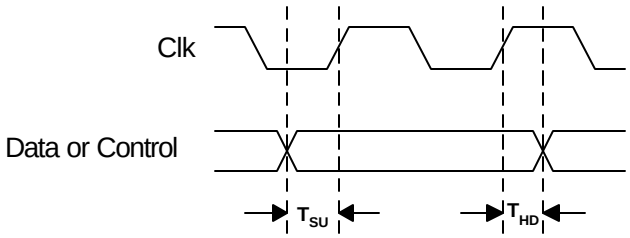
78	Vcc		
79			
80			
81	GND		
82	Int	O8	Interrupt – Asserted when any of the enabled conditions of the IER1 or IER2 are met. De-asserted when the these conditions are cleared by reading the ISR1 and ISR2 registers. The assertion level is dependent upon the setting of PCCR[7].
83	Mode_Change	O8	Mode Change – This signal is asserted whenever the 1284 state machine changes modes by either a 1284 Negotiation sequence or a termination cycle.
84	IMD_Term	O8	Immediate Termination – Asserted with Mode Change to indicate that the mode change was due to an Immediate Termination sequence. This may indicate some type of abnormal operation of the 1284 interface.
85			
86			
87			
88			
89			
90			
91	GND		
92	Vcc		
93			
94			
95			
96			Do Not Use
97			Do Not Use
98			Do Not Use
99			Do Not Use
100			Do Not Use

Table 8 – W91284PIC Pin List

6.0 Timing Diagrams

This section describes the timing and protocol requirements for the client side of the W91284PIC. All timings are provided as the number of Sys_Clk cycles required to meet the minimum or maximum requirements.

The signals on the parallel port side of the chip are asynchronous and are registered and synchronized internally in the W91284PIC. The signals on the client side need to meet the following setup and hold times relative to the system clock.



T_{SU} = 10nS min
T_{HD} = 10nS min

6.1 Register Access

The following diagram illustrates two register reads followed by a register write cycle to any of the control or FIFO registers.

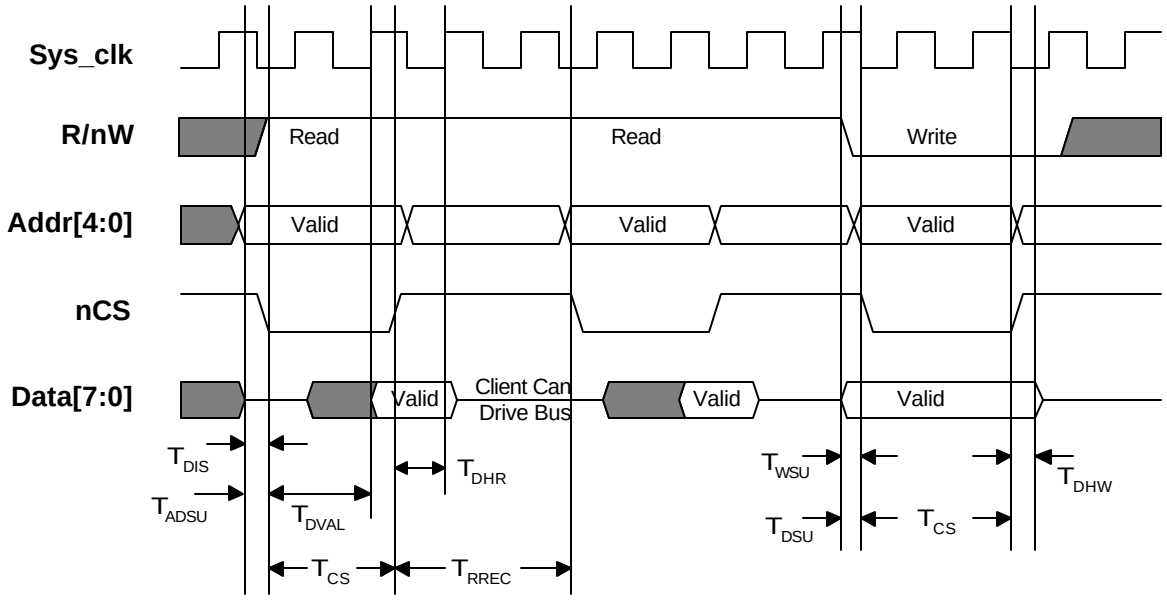


Figure 10 – Register Read/Write Cycle Diagram

Parameter	Min	Max	Description
T_{DIS}	0		Data bus tristate to CS asserted
T_{ADSU}	0		Address valid to CS asserted
T_{DVAL}	1	2	CS asserted to Data valid
T_{CS}	2		CS width
T_{DHW}	0		CS de-asserted to Data not valid, driven
T_{DHR}		1	CS de-asserted to Data not valid
T_{WSU}	0		R/nW low to CS asserted for write cycle
T_{DSU}	0		Data valid to CS asserted
T_{RREC}	2		Recovery time between CS

NOTE: All times are in rising edge of system clocks

Table 9 – Register Read/Write Timing Requirements

6.2 DMA Read (Forward) Data Transfer Cycle

The following diagram illustrates the DMA transfer cycles for a DMA read cycle. These are used when reading data from the receive FIFO in the forward direction.

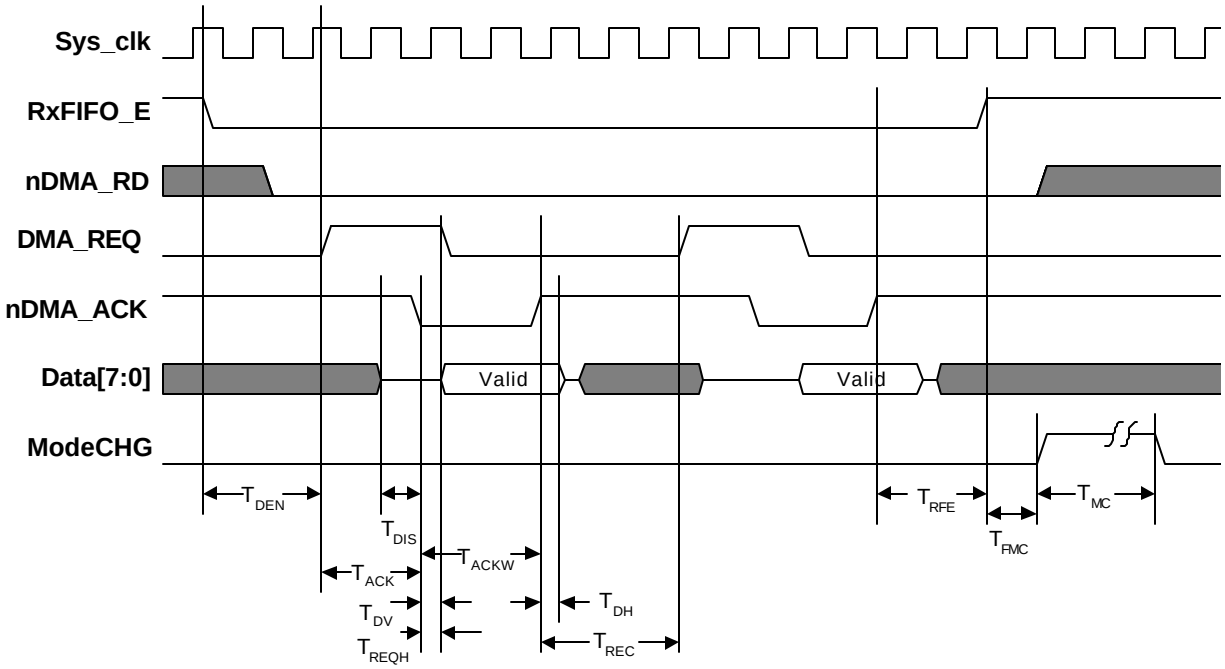


Figure 11 – DMA Read (Forward) Data Transfer

Parameter	Min	Max	Description
T_{DEN}		2	RxFIFO_E low to DMA_REQ
T_{ACK}	1		DMA_REQ to nDMA_ACK asserted
T_{DIS}		0	Data bus tristated to nDMA_ACK asserted
T_{DV}		1	nDMA_ACK to Data valid
T_{REQH}	0	1	nDMA_ACK asserted to DMA_REQ de-asserted
T_{ACKW}	1		nDMA_ACK width
T_{DH}	0	1	Data hold from nDMA_ACK de-asserted
T_{REC}	2	3	DMA recovery – DMA_ACK to DMA_REQ
T_{RFE}	1	2	nDMA_ACK de-asserted to RxFIFO_E asserted
T_{FMC}		1	RxFIFO_E asserted to Mode_Change asserted
T_{MC}	500nS		Mode Change pulse width (TCPR[4:0])

NOTE: All times are in system clocks

Table 10 – DMA Read Timing Requirements

6.3 DMA Write (Reverse) Data Transfer Cycle

The following diagram illustrates the DMA transfer cycles for DMA write cycle. These are used when writing data to the transmit FIFO for the reverse data transfer.

Reverse DMA transfers are ended by clearing the DMA Enable bit, PCCR[2].

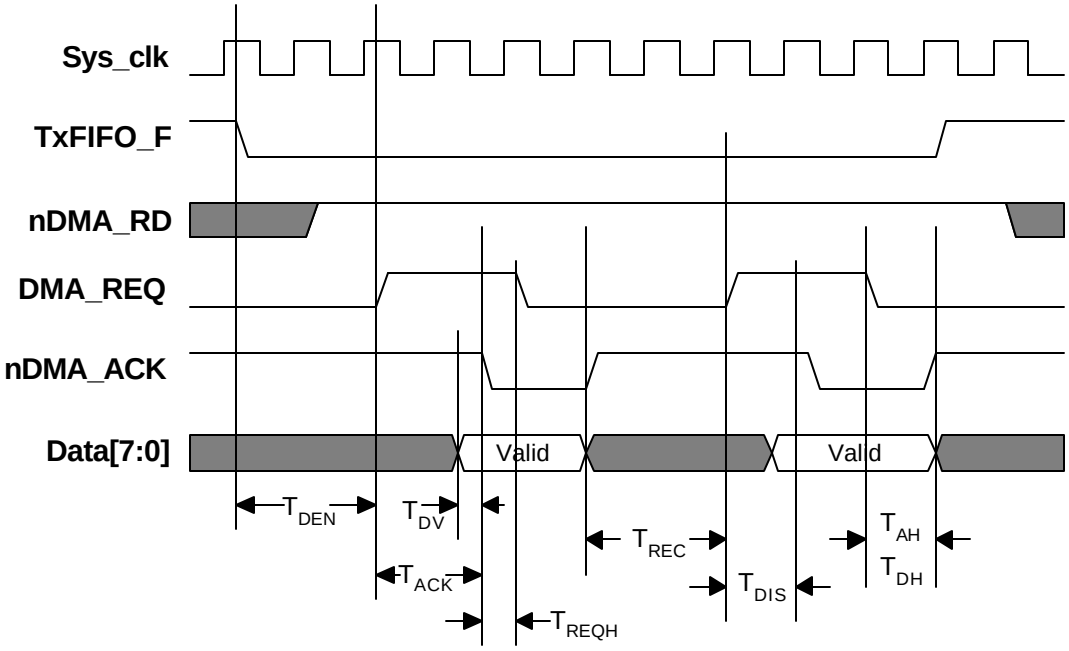


Figure 12 – DMA Write (Reverse) Data Transfer

Parameter	Min	Max	Description
T_{DEN}		2	TxFIFO_F low to DMA_REQ asserted
T_{ACK}	0		DMA_REQ to nDMA_ACK asserted
T_{DV}		0	Data valid to nDMA_ACK asserted
T_{REQH}	1	2	DMA_REQ hold
T_{AH}	1		nDMA_ACK hold from DMA_REQ
T_{DH}	1		Data hold from DMA_REQ de-asserted
T_{REC}	2	3	DMA recovery
T_{DIS}	0	none	Disable DMA - PCCR[2] <= 0

NOTE: All times are in system clock

Table 11 – DMA Reverse Timing Requirements

7.0 Electrical Characteristics

7.1 Absolute Ratings:

Parameter	Symbol	Limits	Unit
DC Supply Voltage	V_{DD}	-0.3 to 7.0	V
Input Voltage	V_{IN}	VDD + 0.3	V
Input Current	I_{IN}	10	µV
Storage Temperature	T_{STG}	-55 to +150	°C

7.2 Recommended Operating Conditions

Parameter	Symbol	Limits	Unit
DC Supply Voltage	V_{DD}	+3 to +5.5	V
Operating Temperature (Ambient)	T_A	0 to +70	°C
Junction Temperature	T_J	<150	°C

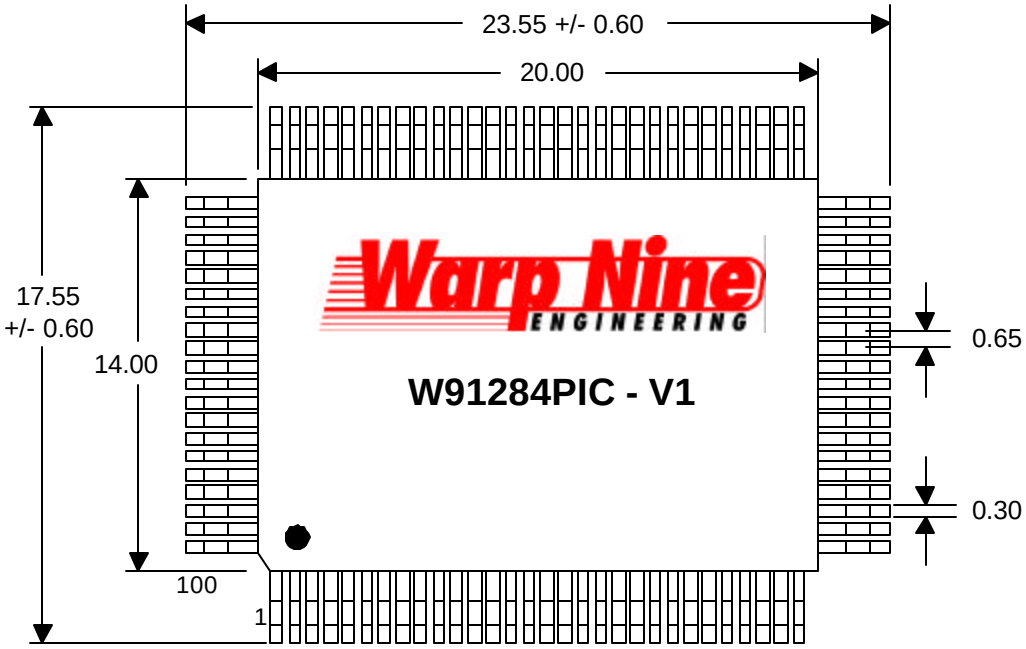
7.3 DC Characteristics

VDD = +5V +/- 5%

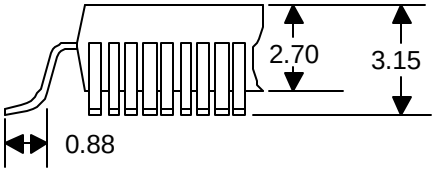
Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{IL}	Voltage Input LOW	TTL			0.8	V
V_{IH}	Voltage Input HIGH	TTL	2.0			V
I_{IN}	Input Current	TTL $V_{IN} = V_{SS}, V_{DD}$	-10	1	10	µA
V_{OH}	Voltage Output HIGH	I_{OH} = Rated current	2.4	4.5		V
V_{OL}	Voltage Output LOW	I_{OL} = Rated Current		0.2	0.4	V
I_{OZ}	TriState Output Leakage	$V_{OH} = V_{SS}$ or V_{DD}	-10	1	10	µA

8.0 Package Definition

The W91284PIC is packaged in an industry standard 100 pin PQFP package.



All dimensions shown in millimeters



9.0 Reference Documents

These documents are required in order to fully understand the interface protocols and usage of the W91284PIC.

9.1 IEEE Standard 1284-1994

"IEEE Standard Signaling Method for a Bi-directional Parallel Peripheral Interface for Personal Computers" IEEE, Document SH17335, Phone 1-908-981-1393

9.2 Draft, IEEE Standard P1284.3

"Standard for Interface and Protocol Extensions to IEEE Std 1284-1994 Compliant Peripherals and Host Adapters", Unreleased. Download from ftp.lexmark.com/pub/ieee/1284.3/d500.pdf.

10.0 Errata

This section outlines any discrepancies between the W91284PIC specification and the shipping part.

1- 5/7/99 Chip Version –V1 Version ID Register (Reg. 1F) – 0x20h

Issue:

Reverse_Request and Nibble mode. If Reverse_Request is set and no data is in the TxFIFO when the chip is negotiated into Nibble mode, then the chip will loop sending the last valid byte that was transmitted. Once data is written into the TxFIFO then valid data will be sent.

Workaround:

The workaround is to not use Reverse_Request with Nibble mode. This bit operates correctly with Byte, ECP and EPP. Also, once negotiated into Nibble mode the host should not set HostBusy (nAutoFd) low until nFault is asserted low. Operating this way causes no problems.

Specification Section:

See section 4.4.1

Severity:

This problem is not severe. The chip operates as defined in IEEE 1284. This will be fixed in a future release of the chip.

2- 5/7/99 Chip Version –V1 Version ID Register (Reg. 1F) – 0x20h

Issue:

BECP mode is not supported. An IEEE negotiation value of 0x18h will not be recognized by this version of the chip.

Workaround:

Bounded ECP may be implemented by using the Reverse_Request bit for ECP mode.

Specification Section:

See section 3.5

Severity:

This problem is not severe. The chip operates as defined in IEEE 1284. This will be fixed in a future release of the chip.