



Macrocad Development Inc.

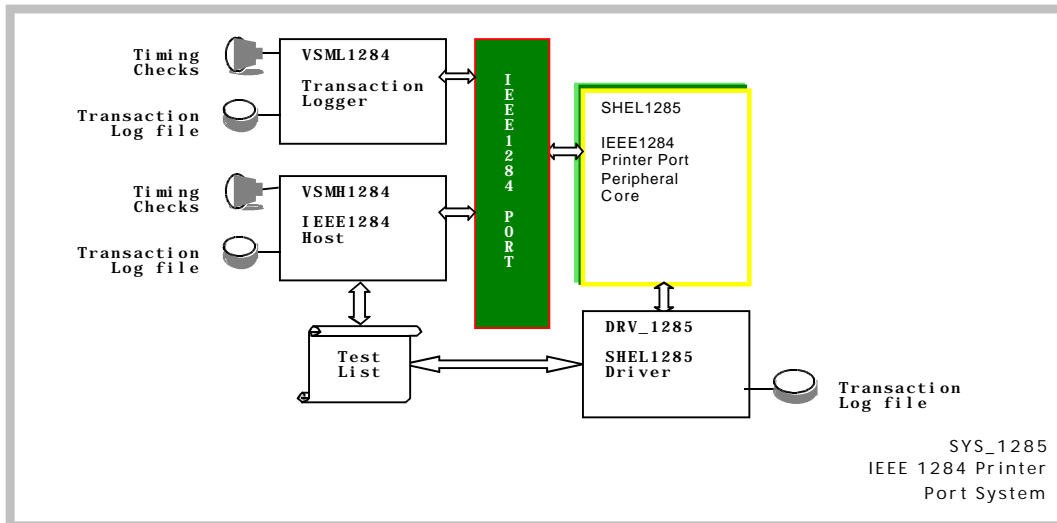
VCM 1285 Parallel Port Model

Description:

Macrocad's VCM 1285 is a synthesizable IEEE1284 compatible RTL behavior model. This model provides the system designer with a drop in peripheral port function for a 1284 parallel printer port. It includes 16 byte buffer memories for improved data communications. The SHEL1285 model contains the bi-directional buffers and the CORE1285. The CORE1285 model contains the functional logic for the 1284 state machines and data buffers. It contains the register set, configuration, DMA, interrupt, and 1284 state machine logic. The VCM1285 can be configured to negotiate for ECP mode on the 1284 port. This is useful when hosts have bursted, (DMA based) transfers scheduled to go to the peripheral application. The VCM1285 is highly configurable. The DMA channel, can be set up via the 1284 interface. The VCM1285 can act as a slave or a master on the peripheral bus.

Features:

- IEEE1284 spec compliant
- ECP, EPP 1.7, EPP 1.9, SPP, PS2, and Buffered SPP modes supported
- DMA operations supported
- Decompression supported
- Peripheral bus master capabilities
- Auto-negotiation to ECP, Byte and Nibble
- 16 deep forward buffer
- 16 deep reverse buffer
- Flexible 1284 port time out features
- Flexible interrupt features
- Digital signal filtering
- Programmable buffer trigger level
- Available in Verilog and VHDL versions
- Test Bench is included
- Well commented, synthesizable RTL code





Architecture:

The VCM peripheral model package contains the following elements:

- 1) IEEE 1284 System Interconnect and Test List
- 2) IEEE 1284 Data Transaction Logger
- 3) IEEE 1284 Virtual Host Model
- 4) VCM 1285 Peripheral Core, Shell Level
- 5) VCM 1285 Peripheral Core Driver

The VCM 1285 shell level model contains the IO buffers, and the core level model. This includes the input, output and bi-directional IO buffers. The core level model contains the peripheral's functional blocks. The signals at the core level and below are all uni-directional.

- 1) IEEE 1284 phase tracker
- 2) IEEE 1284 automatic handshake engine
- 3) Register set
- 4) PPE bus interface
- 5) Timeout and granularity counters
- 6) Dual port RAM buffers and control
- 7) DMA / master engine

The generic application bus is called the PPE bus. Its uni-directional signals are grouped into data, address, and control groups. It has an 8 bit data bus, with 32 bit addressing. The control signals are simple bus request / grant, and transaction request / acknowledge type handshakes. The DMA uses a similar handshake signals.

Pin Descriptions:


VCM1285 – SHEL1285 PORT SIGNALS

IEEE 1284 PRINTER PORT INTERFACE SIGNALS			
Signal Name	Range	Function	Description
pd	7:0	In / Out	Data signals
n_ack		In / Out	Acknowledge from peripheral
busy		In / Out	Peripheral busy
n_fault		In / Out	Peripheral error
perror		In / Out	Paper out / jam
select		In / Out	Peripheral on line
n_strobe		In / Out	Data strobe
n_autofd		In / Out	Auto line feed
n_init		In / Out	Initialize to idle – SPP
n_selectin		In / Out	Select to peripheral
PERIPHERAL application INTERFACE SIGNALS			
Signal Name	Range	Function	Description
o_pped	7:0	Output	Peripheral data from 1284 port
int	1:0	Output	Interrupts
	1		Any bit is set in the int. status register, 'r_sti'
	0		Pulse sets a bit in the int. status register, 'r_sti'
v85_adr	31:0	Output	Peripheral address – master mode
bus_req		Output	Request for peripheral bus
v85_req		Output	Access request to peripheral – master mode
v85_ack		Output	Access acknowledge to peripheral – master mode
v85_wrt		Output	Write access to peripheral
i_pped	7:0	Input	Peripheral data to 1284 port
pointer	4:0	Input	Register selects for peripheral to access register set in VCM1825
bus_ack		Input	Peripheral bus grant
ppe_req		Input	Register access request from peripheral
ppe_ack		Input	Access acknowledge from peripheral
ppe_wrt		Input	Write access
reset		Input	System hardware reset
clk		Input	System clock

Register Map:

Below is a register map which indicates the location of the registers which are accessible through the PPE interface. Each of these registers is 8 bits.



POINTER	OP	REGISTER	DESCRIPTION
00	R	pd	Printer port data pins <read only>
01	R	psr	Printer port pin status <read only>
02	R/W	pcr	Printer port control register
03	R	pin	Printer port control pins <read only>
04	R	ffx	Forward buffer data register <read only>
05	R/W	fea	Feature a register
06	R/W	feb	RESERVED
07	R/W	fei	Interrupt enable register
08	R/W	fem	Master enable register
09	R	exr	Extensibility register <read only>
0a	R/W	ecr	Extended control register
0b	R	sti	Interrupt status register <read only>
0c	W	toc	Time out counter register <write only>
0d	R/W	msk	Mask pin interrupts
0e	R/W	pit	Pin interrupt selects
0f	W	rfx	Reverse buffer data register <write only>
10	R/W	ad0	Master address byte 0
11	R/W	ad1	Master address byte 1
12	R/W	ad2	Master address byte 2
13	R/W	ad3	Master address byte 3
14	R/W	tr0	Transfer count byte 0
15	R/W	tr1	Transfer count byte 1
16	R/W	tr2	Transfer count byte 2
17	R/W	tr3	Transfer count byte 3
18	R/W	id0	ID byte 0
19	R/W	id1	ID byte 1
1a	R/W	grn	Granularity counter
1b	R	sfx	Buffer full status <both buffers>
1c	W	rfa	Reverse buffer address register <write only>
1d	R	eca	Forward address register <read only>
1e	R	pha	IEEE1284 phase tracking
1f	-	----	RESERVED



Register Bit Map:

Below is a more detailed description of each register and the description of each register bit's function or meaning.



REGISTER BIT DESCRIPTIONS		
NAME	BIT	DESCRIPTION
<i>pd</i>	7:0	Printer data pins <read only>
<i>psr</i>	7:4	Port host pins, filtered, status register <read only>
	3	- RESERVED -
	3	N_AUTOFD
	2	N_INIT
	1	N_SELECTIN
	0	N_STROBE
<i>pcr</i>		Port control register This register can directly control the IEEE1284 pin values. Bits 7:3 should be set to '1' before the printer port is enabled, (ecr[0] = '1') to avoid driving IEEE1284 pins during initialization. If no modes are enabled, (fem[6:2]), then the IEEE1284 pins can be controlled by this register.
	7	BUSY
	6	N_ACK
	5	PERROR
	4	SELECT
	3	N_FAULT
	2:0	- Reserved -
<i>pin</i>		Port peripheral pins, filtered, status register <read only>
	7	BUSY
	6	N_ACK
	5	PERROR
	4	SELECT
	3	N_FAULT
	2:0	- RESERVED -
<i>ffx</i>	7:0	Forward Buffer Data <read only>
<i>fea</i>		Feature A register This register enables buffer trigger levels, soft reset for the core, reset for buffer pointers, decompression, and printer port operations.
	7:6	11 : interrupt when forward buffer is more than ¾ full 10 : interrupt when forward buffer is more than ½ full 01 : interrupt when forward buffer is more than ¼ full 00 : interrupt when forward buffer is more than empty
	5:4	11 : interrupt when reverse buffer is less than ¼ full 10 : interrupt when reverse buffer is less than ½ full 01 : interrupt when reverse buffer is less than ¾ full 00 : interrupt when reverse buffer is less than full
	3	Software Reset <write only>
	2	Reset Buffer Pointers <write only>
	1	Decompression enabled
	0	Printer port enable Setting this bit to 0, will force the IEEE1284 pin core outputs to high impedance.
<i>feb</i>	7:0	Feature b register This register enables additional core features.
	7:1	- RESERVED -
	0	Use buffer for reverse ID transfers, not ID0
<i>fei</i>		Interrupt enable register This register enables an interrupt to be generated on the selected conditions.
	7	FIFO interrupt enable
	6	Timeout detect interrupt enable
	5	Pin Select Interrupt enable
	4	ECP mode Address forward transfer detect enable
	3	Printer reset interrupt enable
	2	Transfer complete interrupt enable <master mode>
	1	Negotiation start detect interrupt enable
	0	Transfer start detect interrupt enable
<i>fem</i>		Master enable register This register enables the various IEEE1284 modes, and automatic transfer modes.



REGISTER BIT DESCRIPTIONS		
NAME	BIT	DESCRIPTION
	7	EPP enable
	6	ECP enable
	5	SPP - PS2 enable
	4	Auto transfer enable
	3	Master address auto increment
	2	Auto-negotiate enable
	1:0	- RESERVED -
<i>exr</i>	7:0	Extensibility byte requested by host
<i>ecr</i>		Extended control register
	7	Open drain drive disable on printer port
	6	Enable reverse request if buffer is not empty
	5	FIFO disable
	4	Global interrupt mask bit
	3	Master enable
	2	Decompress incoming data after buffer
	1:0	- RESERVED -
<i>sti</i>		Interrupt status register
	7	FIFO interrupt
	6	Timeout detect interrupt
	5	Pin select interrupt
	4	Spp mode transfer complete interrupt
	3	Printer reset interrupt
	2	Transfer complete interrupt <master mode>
	1	Negotiation start detect interrupt
	0	Transfer start detect interrupt
<i>toc</i>	7:0	Interrupt timeout counter <write only> Writing to this location will set the time out counter to this value The timeout counter will count by 1 each time the granularity counter cycles. The granularity counter is a module, or cyclical counter. When the timeout counter expires, or reaches 0x00, then the timeout interrupt can be generated, (if enabled with fei[6]). Setting this register to 0x00 will disable the interrupt generation.
<i>msk</i>		Mask pin interrupt
	7	n_autofd edge detect enable
	6	n_init edge detect enable
	5	n_selectin edge detect enable
	4	n_strobe edge detect enable
	3	n_autofd level enable
	2	n_init level enable
	1	n_selectin level enable
	0	n_strobe level enable
<i>pit</i>		Mask pin interrupt
	7	n_autofd edge detect, 1= rising edge, 0= falling edge
	6	n_init edge detect, 1= rising edge, 0= falling edge
	5	n_selectin edge detect, 1= rising edge, 0= falling edge
	4	n_strobe edge detect, 1= rising edge, 0= falling edge
	3	n_autofd level
	2	n_init level
	1	n_selectin level
	0	n_strobe level
<i>rfx</i>	7:0	Reverse buffer Data <write only>
<i>ad0</i>	7:0	Address register byte 0 <least significant>
<i>ad1</i>	15:8	Address register byte 1
<i>ad2</i>	23:16	Address register byte 2
<i>ad3</i>	31:24	Address register byte 3 <most significant>
<i>tr0</i>	7:0	Transfer register byte 0 <least significant>
<i>tr1</i>	15:8	Transfer register byte 1



REGISTER BIT DESCRIPTIONS		
NAME	BIT	DESCRIPTION
<i>tr2</i>	23:16	Transfer register byte 2
<i>tr3</i>	31:24	Transfer register byte 3 < most significant >
<i>id0</i>	7:0	ID register byte 0 < least significant >
<i>id1</i>	15:8	ID register byte 1 < most significant >
<i>grn</i>	7:0	Granularity of Port operations for auto modes < clock divisor > This register will determine the modulo value for the granularity counter. The granularity counter determines the time between signal changes for signals driven by the peripheral on the IEEE1284 bus, when in automatic modes. Thus, if the clock period was 20ns, and the granularity counter was set to 0x09, then the fastest signal transitions for the IEEE1284 peripheral driven pins would be 500 ns.
<i>sfx</i>		Buffer full status
	7	Forward buffer full
	6	Forward buffer more than 3 / 4 full
	5	Forward buffer more than 1 / 4 full
	4	Forward buffer not empty
	3	Reverse buffer full
	2	Reverse buffer more than 3 / 4 full
	1	Reverse buffer more than 1 / 4 full
0	Reverse buffer not empty	
<i>rfa</i>	7:0	Reverse buffer address < write only >
<i>eca</i>	7:0	Forward address < read only >
<i>pha</i>	7:0	IEEE1284 phase tracking < read only > The phases are shown below 0x00 - spp forward idle 0x01 - spp forward data transfer 0x0f - spp reset - initialize 0x14 - negotiate phase 0x18 - terminate phase 0x24 - nibble/byte idle 0x26 - nibble/byte reverse data transfer 0x28 - nibble/byte host busy data not available 0x2c - nibble/byte host busy data available 0x2e - nibble/byte interrupt host 0x3f - ecp set up phase 0x30 - ecp forward idle 0x31 - ecp forward data transfer 0x34 - ecp reverse idle 0x36 - ecp reverse data transfer 0x38 - ecp host recovery 0x3e - ecp forward to reverse phase transition 0x3c - ecp reverse to forward phase transition 0x40 - epp idle 0x49 - epp forward address transfer - write 0x48 - epp forward data transfer - read 0x41 - epp forward address transfer - write 0x42 - epp forward data transfer - read 0x4f - epp interrupt host



Interrupts:

If enabled, an interrupt can be generated in the following ways:

- 1) forward phase - buffer service required <buffer reached trigger level>
- 2) reverse phase - buffer service required <buffer empty>
- 3) interrupt from the timeout counter
- 4) selected pin activity combinations <programmable combinations>
- 5) ECP address detected from host <forward mode>
- 6) interrupt from a printer reset
- 7) interrupt at end of forward transfer <master DMA mode transfer complete>
- 8) spp forward phase – rising edge of SELECTIN detected <negotiation start>
- 9) spp forward phase – falling edge of STROBE detected <transfer start>

Interrupts are enabled by clearing (arming) the interrupt service bit.

The sti register shows the status of the interrupts, and the fei provides the enables for each of the 8 interrupts.

Interrupts based on specific IEEE1284 signal activity can be accomplished with the interrupt mask (msk) registers. In this case any rising or falling edge for each of the 4 host driven IEEE1284 signals can be selected. Either high or low levels can also be selected.

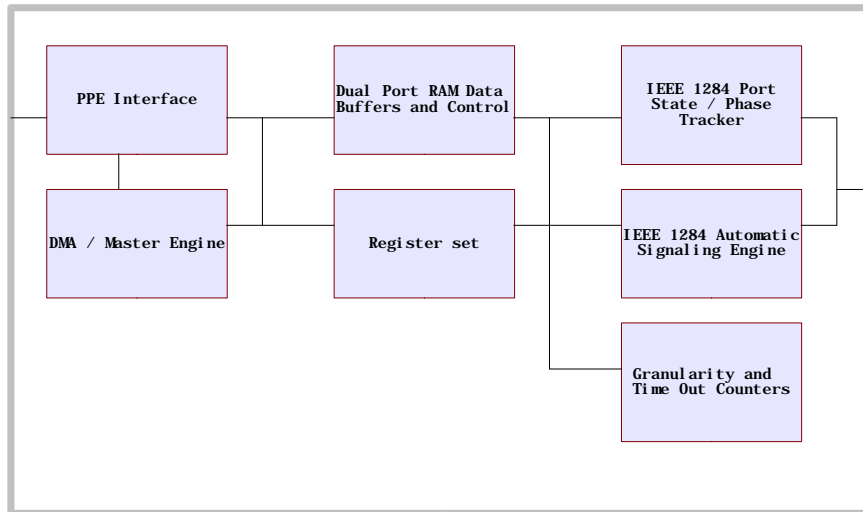
Functional Blocks:

The VCM1285 core level model is comprised of the following functional blocks:

- 1) IEEE 1284 PHASE TRACKER
The phase tracker is needed for automatic transfers. It detects changes in the state of the IEEE1284 port, and generates the port phase and direction from that information
- 2) IEEE 1284 AUTOMATIC HANDSHAKE ENGINE
This function is responsible for automatic negotiation, and transfers. The state machines which handle these automatic transactions are programmable by the peripheral application.
- 3) REGISTER SET
The register set contains the control, status, and data registers for the core.
- 4) PPE BUS INTERFACE
The PPE interface provides a means to access the register set through the PPE interface bus. These registers contain information about the state of the IEEE1284 port, the automatic state machine operations, and control of the transactions to and from the IEEE1284 port.
- 5) TIMEOUT AND GRANULARITY COUNTERS
These counters provide timing for the IEEE 1284 port signaling engine, and also provide a means to avoid lockups on the port when in EPP mode.
- 6) DUAL PORT RAM BUFFERS AND CONTROL
The transfers to and from the IEEE 1284 port require data buffers in each direction. This is especially true when using master mode, and automatic transfer features. These 2 buffers are 9 bits wide.
- 7) DMA / MASTER ENGINE
The master feature provides a way to automatically transfer data by requesting access to



the PPE bus, and moving data to and from the dual port RAM buffers. The master engine will request the bus, and transfer the data when the buffer status indicates that the reverse buffer is empty, or the forward buffer has reached a trigger level.



Peripheral Application Interface:

Slave mode:

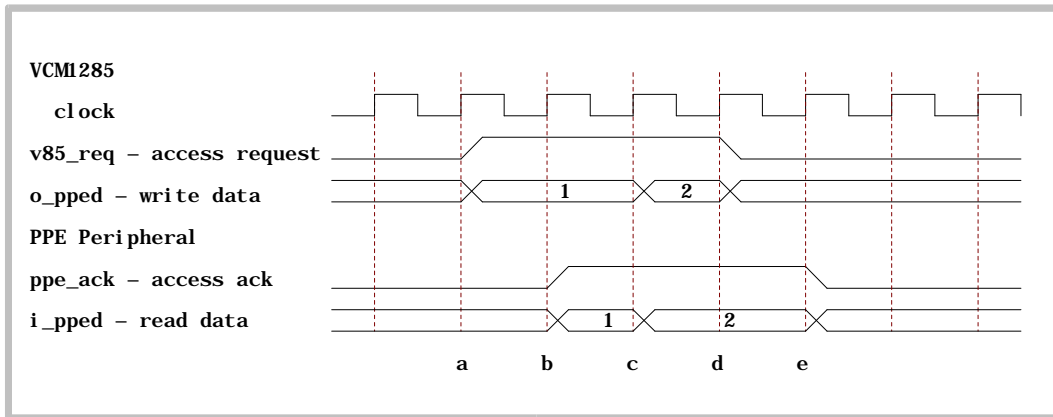
The VCM1285 model will transfer data to and from the PPE interface with a simple synchronous handshake. When the PPE_REQ signal is activated, the VCM1285 will check the register pointer and the type of access (read or write) to determine the action to be taken. It will (in most cases) assert the V85_ACK signal to indicate a completed register access. In the figure below, the request, write data and register index (address) are asserted after clock edge "a". On clock edge "b", this is detected by the peripheral core, and a response is asserted. On the clock edge when the request and response are asserted, a register transfer takes place. In this case clock edges "c" and "d" have register write transfers. If the transfers were reads, then the read data would be on clock edges "c", "d" and "e". The register value on clock edge "e" would be the same as clock edge "d".

Master mode:

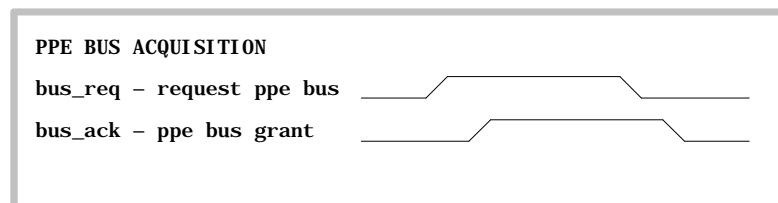
The VCM1285 can also operate in master mode on the PPE interface. In this mode, the VCM1285 must first acquire bus ownership. A simple request / acknowledge handshake is used. When the ownership has been established, the transfer can proceed. The VCM1285 activates the V85_REQ signal and waits for the PPE_ACK signal, indicating completion of the access. The VCM1285 will request bus access when master mode is enabled, (r_ecr[3] = 1) and the fifo is not empty. This feature only works in the forward phase, (data transmitted from host to peripheral). When the bus is in master mode, and a transfer happens on the PPE interface, the address and transfer counters are updated. The 32 bit address counter will increment, and the 32 bit transfer counter will decrement. When this transfer counter reaches zero, the address and transfer counters are frozen and will no longer be updated, even when transfers are still taking



place on the PPE interface in master mode. An interrupt can be generated when the transfer count has reached its terminus, (zero).



The peripheral bus will be granted until the current owner drops its request. This is a very simple type of bus arbitration, and it requires some software discipline to avoid lockouts. The clock is not shown in this diagram, but the signals are assumed to be synchronous.



IEEE 1284 Port Transfer Operations:

Refer to the IEEE1284 specification for the basics of parallel printer port transfers as defined by the IEEE1284 standard. This standard includes the SPP modes, (0=standard Centronics, 1=PS2, 2=standard FIFO), ECP, and EPP modes of transferring data on this parallel interface.

IEEE1284 FORWARD TRANSFERS

In the IEEE1284 forward phase, (data transferred FROM the IEEE1284 host TO the IEEE1284 peripheral), the VCM1285 will transfer data into the data buffer. This is the register location which the PPE application reads in response to IEEE1284 transfers.

IEEE1284 REVERSE TRANSFERS

In the case of reverse phase, a separate buffer will be written to by the PPE application. This data will be available for transfer across the IEEE1284 port to the IEEE1284 host.

The IEEE1284 printer port interface has a number of modes and operational phases. Below is a typical sequence of phases for doing a transaction in ECP mode.



MODE	PHASE
SPP	forward
SPP	negotiate
ECP	setup
ECP	forward
ECP	reverse
ECP	forward
ECP	host recovery
ECP	forward
ECP	terminate
SPP	forward

With the exception of the transition from negotiate to setup, the ECP operational phases must transition into and out of forward mode.

PROGRAMMING STEPS FOR REVERSE ID

Program the core for automatic negotiation to reverse byte mode transfers.

```

PCR = 0xff
ECR = 0x80
FEA = 0xd1
FEB = 0x00
FEI = 0x02
FEM = 0x34
GRN = 0x80
  
```

The PCR register will be set so that the IEEE1284 pins are not driven by this register directly. The ECR register has the open drain feature for the IEEE1284 pins disabled. The FEA register sets the fifo interrupt levels, and enables the core printer port pins. FEI sets the interrupt to happen when negotiation start is detected. FEM is set to enable SPP mode, auto transfer, and auto negotiate.

At this point, the host can negotiate to reverse ID mode, and the core will generate an interrupt when negotiation start is detected. There is no need for the programming entity to intervene, since the core will negotiate, and transfer the ID data with no further programming steps required. This also includes the termination steps.

PROGRAMMING STEPS FOR ECP TRANSFERS

Program the core for automatic negotiation to ECP mode transfers.

```

PCR = 0xff
ECR = 0x80
FEA = 0xf3
FEB = 0x00
FEI = 0x02
FEM = 0x54
GRN = 0x24
  
```

The PCR register will be set so that the IEEE1284 pins are not driven by this register directly. The ECR register has the open drain feature for the IEEE1284 pins disabled. The FEA register sets the fifo interrupt levels, and enables the core printer port pins. FEI sets the interrupt to



happen when negotiation start is detected. FEM is set to enable SPP mode, auto transfer, and auto negotiate.

At this point, when the host negotiates to ECP mode, the 1285 core will generate an interrupt when negotiation start is detected. The peripheral can now check the PHA register to determine the phase of the IEEE1284 bus. This register should indicate the progress through negotiation, to ECP setup to ECP idle, then ECP forward mode. The EXR register will indicate the requested mode when negotiation is complete. Alternately, the core can set the FEI register interrupt mask to include the FIFO interrupt, so that after the negotiation start interrupt, the next interrupt should be the forward fifo trigger interrupt, indicating the fifo should be drained.

In addition to the other interrupts, one can use the timeout interrupt counter to present an interrupt if there is not an interrupt due to failed negotiation, or some other unforeseen occurrence. When an expected interrupt is received, it is important to set the timeout counter to 0x00 to disable it to avoid getting a spurious interrupt.

Transfer Methods:

There are several ways to interface with the core1285 register set to transfer data. For all modes, data is always transferred through the buffer registers ffx for forward transfers from the host to the peripheral, and the rfx buffer for reverse transfers from the peripheral to the host.

Manual mode:

The individual signal pins of the core1285 model can be manipulated by the peripheral application directly via the pcr, pin, and psr registers. Together with flexible interrupt capabilities, this can be an effective way to handle non-bursty transfers of data. This method requires polling of the 1284 port status, or reliance on interrupts to properly execute the signal handshakes for transfers on the 1284 port.

Automatic mode:

In the automatic mode, the fea, fem, ecr, and sfx registers are set up prior to or monitored during a transfer when the application knows what transfer is going to be done. This case works well for reverse modes, when the application knows when it has data to transfer to the host. The application may not always know when the host will have data to transfer to the peripheral, so there are interrupts which can alert the peripheral application to an incoming forward transfer. In reverse direction, the data transferred can either be data from the reverse buffer, or from the ID0 register. This is determined during the negotiation phase.

Semi-automatic mode:

In this mode, the application may use interrupts in manual mode to negotiate, or handle unforeseen forward transfers, and once a transfer has been set up, then use automatic mode to transfer the data.

DMA Master mode:

In this mode, the automatic transfer capability is used in conjunction with the DMA master capability to transfer larger amounts of data with little peripheral application intervention. The transfers are done automatically, and the buffer is serviced by the DMA function to keep larger



block transfers going into a buffer outside the VCM1285 model. The ad 0:3 and tr 0:3 registers are used for controlling this DMA transfer.

Additional Features:

The incoming IEEE1284 port host driven signals are filtered to ignore glitches, and synchronized to the peripheral clock at the same time.

Peripheral driven signals can be either open drain, (sink only) or driven as source-sink drivers.

Buffers are implemented as dual port rams for flexibility and expandability.

Buffer interrupt levels are selectable. In the fea register, bits 7:6 select the forward interrupt trigger level. Bits 5:4 select the reverse interrupt trigger level.

A timeout counter (toc) is provided when an indeterminate number of bytes are transferred in the forward direction (host to application). In this case, the transfer may not fill the buffer to the trigger level point, and thus no interrupt will alert the application to the end of the transfer. The timeout counter can provide a safety net so that an interrupt will occur if the timeout counter counts down to 0x00. The timeout counter is reset when there is a transfer of data in the forward direction.

There are two options for decompression in the forward direction. Decompression can occur prior to data being written to the buffer, or the data and decompression information is written to the buffer, and decompressed when the application reads the data from the buffer, (this can also work in DMA Master modes of operation).

Reference Documents:

DESCRIPTION	SOURCE	NAME
IEEE1284 specification	IEEE	1284-1994 IEEE Standard Signaling Method for a Bidirectional Parallel Peripheral Interface for Personal Computers (ANSI)
ISA implementation	Microsoft, Hewlett Packard	"The IEEE 1284 Extended Capabilities Port Protocol and ISA Interface Standard"