Semiconductors

UM10053_1

PDIUSBD12 USB EPP Eval Kit

Rev. 01.00 — 12 May 2004 User manual

Note: File name of the previous revision was D12EPP_Manu_Ver1.0.pdf.

Document information

Info	*Content*
Keywords	USB, PDIUSBD12, D12, EPP
Abstract	The USB-EPP eval kit is a comprehensive kit that enables you to discover the full potential of the Philips PDIUSBD12. It provides the know-how on converting existing EPP into USB devices.

PHILIPS

Revision history

Rev	Date	Description
1.0	April 2004	Changed the file name to UM10053_1 Updated the main board schematics Updated the code.
Pre-1.0	Nov 1998	First release

Contact information

For additional information, please visit: *http://www.semiconductors.philips.com/*
For sales office addresses, please send an email to: *sales.addresses@www.semiconductors.philips.com*

This is a legal agreement between you (either an individual or an entity) and Philips Semiconductors. By accepting this product, you indicate your agreement to the disclaimer specified herein.

DISCLAIMER

PRODUCT IS DEEMED ACCEPTED BY RECIPIENT. THE PRODUCT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, PHILIPS SEMICONDUCTORS FURTHER DISCLAIMS ALL WARRANTIES, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANT ABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THE ENTIRE RISK ARISING OUT OF THE USE OR PERFORMANCE OF THE PRODUCT AND DOCUMENTATION REMAINS WITH THE RECIPIENT. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL PHILIPS SEMICONDUCTORS OR ITS SUPPLIERS BE LIABLE FOR ANY CONSEQUENTIAL, INCIDENTAL, DIRECT, INDIRECT, SPECIAL, PUNITIVE, OR OTHER DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THIS AGREEMENT OR THE USE OF OR INABILITY TO USE THE PRODUCT, EVEN IF PHILIPS SEMICONDUCTORS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Content

1. INTRODUCTION 6
2. SETTING UP THE USB-EPP EVAL SYSTEM 6
2.1. Installing the hardware 7
2.2. Installing the software 8
2.3. Running the applet 10
2.3.1. Description of D12 endpoint configuration 11
2.3.2. Description of D12 Endpoints 11
3. HARDWARE AND FUNCTION OVERVIEW 13
3.1. Main board block diagram 13
3.2. Daughter board block diagram 14
3.3. Description of the CPLD state machine 14
3.4. Description of registers 15
3.4.1. Address mapping of CPLD internal registers and D12 registers 15
3.4.2. Description of the daughter board registers 16
3.5. Signals description in the EPP mode 17
3.6. Miscellaneous 18
4. APPENDIX 18
4.1. Appendix A: Placement of the components of main and daughter board 18
4.2. Appendix B: Descriptions of the connectors and jumpers 19
4.2.1. Microcontroller connectors J5 and J6 20
4.2.2. Test header J9 21
4.2.3. ISP Connector J4 22
4.2.4. USB serials-B connector J8 (upstream) 23
4.2.5. RS-232 interface connector J3 23
4.2.6. Jumper J1 23
4.2.7. Parallel Port connector and its test header 24
4.2.8. IEEE1284—A connector description 25
4.3. Appendix C: Main and daughter board schematics 26
4.3.1. Main board schematic 26
4.3.2. Daughter board schematic 26
4.4. Appendix D: Bill of materials (BOM) 29
4.4.1. Main board BOM 29
4.4.2. Daughter Board BOM 31
4.5. Appendix E: CPLD VHDL simulation waveforms 32

4.6. Appendix F: CPLD VHDL source codes 36
4.7. Appendix G: PAL (16L8) equations for the main board 40

1. Introduction

The Philips USB-EPP evaluation (eval) kit enables you to discover the full potential of Philips PDIUSBD12 (or D12). It also offers you the know-how to convert existing Enhanced Parallel Port (EPP) devices into USB devices. The kit comes with the D12, main (or evaluation) and daughter boards, test application program (or applet), USB driver, and sample firmware source codes.

The D12 is a high-performance USB peripheral controller that not only offers DMA transfer capability, but also features for a cost-effective microcontroller-based system. This kit gives you the opportunity to perform a thorough evaluation of the capabilities and features of this device as well as to quickly and easily realize the technical know-how to convert existing EPP devices, such as digital still cameras, mass storage devices, and scanners into USB devices. The daughter board, which is a simple memory board with EPP port interface, is used to emulate an EPP device, such as an EPP scanner during the eval test.

To run this kit, you only require a new generation PC (motherboard with USB port) with Microsoft® Windows® 98 operating system. The firmware provided with the kit is written in the C language, thereby allowing you to port it to any other platforms for compiling. With this kit, you can develop your USB devices using the firmware and hardware schematics.

2. Setting up the USB-EPP eval system

To set up the USB-EPP eval system using the kit, connect the USB-EPP main and daughter boards to the host system as shown in Figure 2-1, and then install the software provided with the kit. Before you set up such a system, however, check to ensure that you have the following:

- Host system with USB motherboard or USB plug-in card;
- Microsoft Windows 98 operating system; and
- USB-EPP main and daughter boards, USB cable, and diskette provided with the kit.

Depending on your preference, you may need the optional RS-232 cable with a 9-pin Male and Female D-type connectors for the bus-powered mode and the AC-DC power adapter with 5 V output for the self-powered mode.

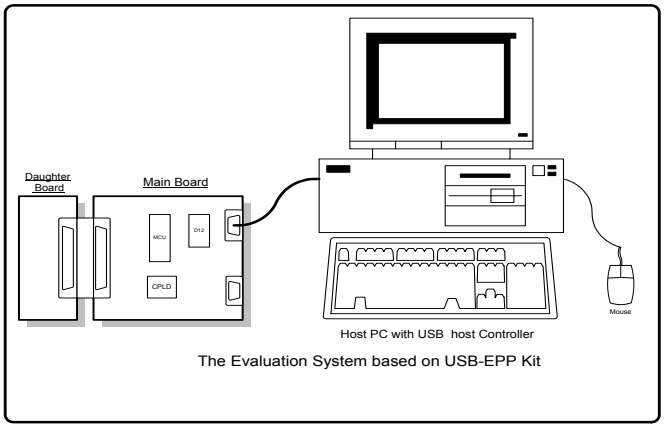

Figure 2-1: USB-EPP eval system

2.1. Installing the hardware

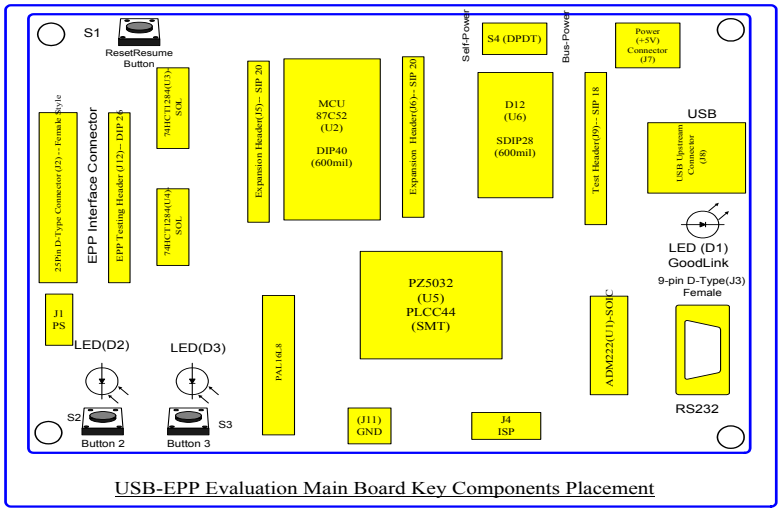

Figure 2-2: Locations of the components on the main board

Figure 2-2 shows you the various components on the main board.

Installing the hardware:

1. Make a backup copy of the original diskette provided with the kit.
2. Short pin 1 and pin 2 of J1 by placing a jumper over them so that the daughter board can receive power supply.
3. If the main board is used for connecting to EPP devices, such as the scanner, then short pin 2 and pin 3 of J1 by placing a jumper over them.

4. Connect the daughter board to the main board through the 25-pin D-type connector.
5. If you are using the self-powered mode, you will need to provide 5 V DC supply to the power connector (J7) from an AC-DC power adapter with at least 500 mA output. Thereafter, flip switch S4 to the direction with label self-power on the main board.
6. If you are using the bus-powered mode, flip switch S4 to the direction with label bus-power on the main board.
7. Connect one end of the USB cable to the upstream connector (J8) on the main board and the other end to the host system or USB hub.
8. Once the USB cable is connected, follow the instructions on the screen to install the device driver.

2.2. Installing the software

The software responsible for controlling the hardware and ensuring that it works includes the firmware, the test application program (or the applet) and device drivers.

The firmware is already stored in the 87C52 microcontroller (OTP type). If you want to change the firmware or use your own firmware, remove the 87C52 microcontroller and replace it with your own microcontroller.

Installing the applet:

1. Insert the diskette into your disk drive.
2. Copy file D12Test.exe contained in the diskette to a directory of your choice. For example: C:\USBEPP.
3. Create a shortcut by dragging file D12Test.exe from the directory to the desktop. An icon similar to that in Figure 2-3 will appear on the desktop.

Figure 2-3: D12Test.exe icon on the desktop

Installing device drivers:

1. When you connect the USB cable to your host system for the first time, a dialog box similar to Figure 2-4 appears. Click ***Next***[1].

[1] In this document, items that you type or click are indicated in ***bold***.

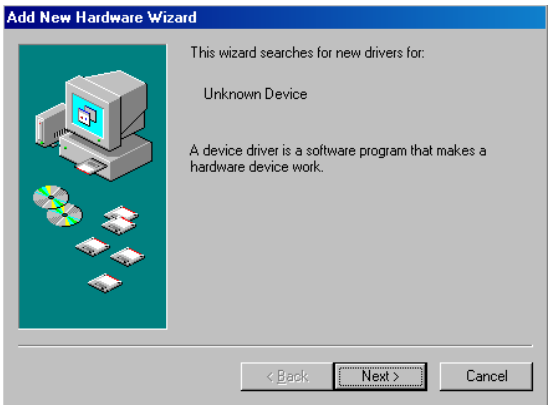

Figure 2-4: Add New Hardware Wizard

2. When a dialog box similar to Figure 2-5 appears, select the first item and click ***Next***.

Figure 2-5: Add New Hardware Wizard

3. When a dialog box similar to Figure 2-6 appears, select ***Floppy disk drives*** and click ***Next***.

Figure 2-6: Add New Hardware Wizard

4. When a dialog box similar to Figure 2-7 appears, click ***Next***.

Figure 2-7: Add New Hardware Wizard

5. When a dialog box similar to Figure 2-8 appears, click ***Finish*** to end the installation.

Figure 2-8: Add New Hardware Wizard

2.3. Running the applet

The applet supports three test modes: Print, Scan and Loopback. The print and scan modes allow the eval board to emulate the printer or scanner environment. The Loopback mode can be used to demonstrate the integrity of data when a large data packet is sent or received from the host system.

To run the applet, click on the ***D12Test.exe icon*** on your desktop. An interface similar to Figure 2-9 appears.

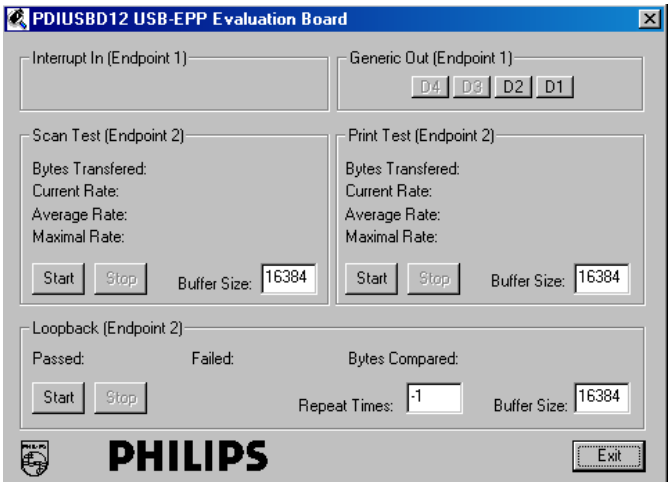

Figure 2-9: Applet

2.3.1. Description of D12 endpoint configuration

Table 2-1 describes the PDIUSBD12 endpoints configuration mode 0, that is, the nonisochronous mode. The reset 3 mode (the isochronous mode) can be found in the D12 specification.

Table 2-1: PDIUSBD12 endpoint configuration

Endpoint number	Endpoint index	Transfer type	Endpoint type	Direction	MaxPacketSize (bytes)
0	0	Control OUT	Default	OUT	16
	1	Control IN		IN	16
1	2	Generic OUT	Generic	OUT	16
	3	Generic IN	Generic	IN	16
2	4	Generic (Main) OUT	Generic	OUT	64 (another 64 for double buffer)
	5	Generic (Main) IN	Generic	IN	64 (another 64 for double buffer)

2.3.2. Description of D12 Endpoints

Table 2-2 describes in detail the operations of the endpoints.

Table 2-2: Operation of endpoints

Endpoint number	Endpoint type	Operations
1	Generic IN	This pipe is defined as the interrupt pipe. The USB-EPP eval board sends specific data packet to the host system when the test key is pressed or released.
1	Generic OUT	This pipe is defined as the Bulk Out pipe. The data packet received from the host system is interpreted as LED control, and the firmware will light up the corresponding LED.
2	Main IN Main OUT	These pipes are defined as the Bulk IN or OUT endpoint. The applet and the eval board support three test modes: loopback, print and scan.

The Generic IN and Generic OUT endpoints have a maximal packet size of 16 bytes, thereby making them suitable for devices that require small size data transfer such as the keyboard, mouse and logic controls. The main endpoints have a maximal packet size of 64 bytes (for the Bulk or isochnorous mode) or 128 bytes (for the isochnorous mode) with double buffering capacity. Therefore, they are suitable for high data rate and large size data transfer.

The three test modes that are supported on main endpoints are:

- *Scan mode*: In this mode, the eval board emulates the scanner environment. This mode is used to evaluate the maximal Bulk IN transfer rate.
- *Print mode*: In this mode, the eval board emulates the printer environment. This mode is used to evaluate the maximal Bulk OUT transfer rate.
- *Loopback mode*: In this mode, the eval board receives data packets from the Main OUT endpoint (Endpoint Index 4) and sends them back to the host system from the Main IN endpoint (Endpoint Index 5). This mode is used to test the ability of the firmware to control the data flow and also the integrity of data transfers.

The host system can inform the firmware to set any possible modes because the firmware is protocol based. Therefore, there is no longer any need to physically set the scan test mode before running the scan test on the firmware running on the eval board.

The “buffer size” of the applet is the size of the data buffer that the applet passes to the USB system drivers for receiving or transmitting. It is therefore the responsibility of the USB system drivers to divide them into smaller data packets for example, 64 bytes for bulk transfer. The USB system drivers, however, limit the maximal buffer size. The best transfer rate over USB can be achieved by optimizing the buffer size.

The buffer size in the Loopback test mode is also limited by the USB-EPP DMA burst transfer size (the DMA_BTS register of CPLD is only 14 bits, that is, 16 kbytes) on the device side. As in the DMA mode, CPLD needs to send the EOT_N signal to D12 when the DMA is receiving and transmitting the maximal buffer size, which is determined by the CPLD internal register DMA_BTS. Since the daughter board has 512 kbytes SRAM, it would need to store 16 kbytes in 32 blocks of memory. The default buffer size (maximal size in the DMA mode) for the Loopback test is 16384 bytes. If the infinite loopback mode is to be used, “-1” should be set the in the Repeat Times box.

3. Hardware and function overview

3.1. Main board block diagram

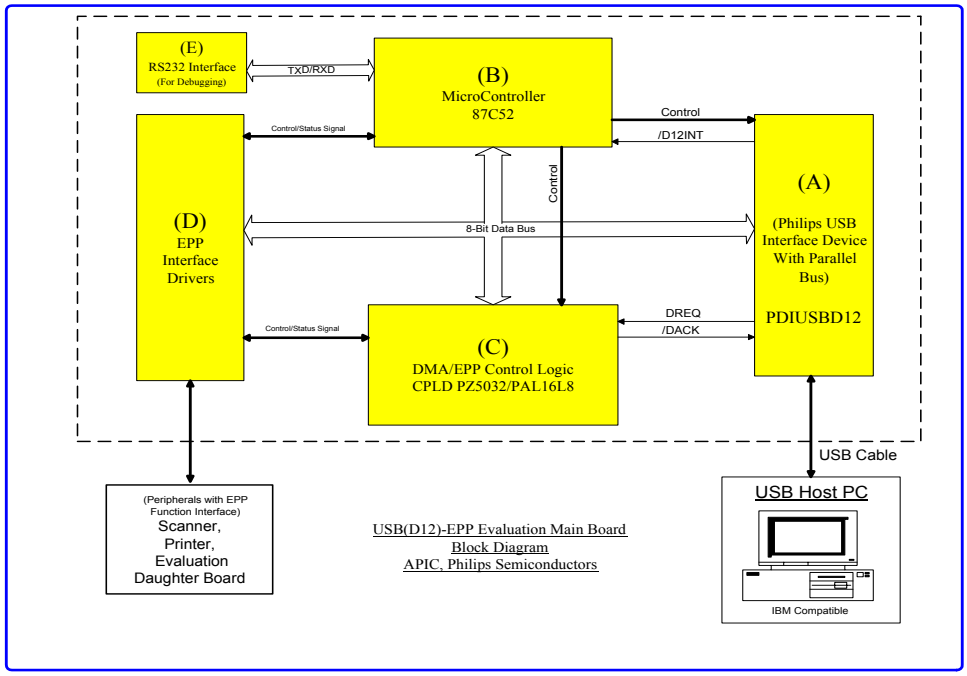

Figure 3-1: Main board block diagram

There are five elements on the main board. The board has three interfaces: the USB interface that is connected to the host system, the EPP interface that is connected to peripherals such as printer and scanner, and the RS-232 interface that is connected to the host system for firmware debugging only.

Element (A) is a Philips PDIUSBD12 USB interface device. It implements all the functions of the USB device, and has an 8-bit parallel data bus for microcontroller (B) or DMA access (C).

Element (B) is a microcontroller. It is a member of the 8051 family. In this case, the Philips 87C52 is used. It implements USB enumeration, EPP negotiation, D12 interrupt service, DMA transfer management, and RS-232 port monitoring.

Element (C) is the DMA and EPP control logic. It is implemented by CPLD (Philips PZ5032) and PAL16L8. This element supports D12 DMA and EPP access timing as well as a built-in 14-bit counter for the DMA burst control.

Element (D) performs only a pure buffering function for the EPP interface. It consists of 2pcs 74HCT1284.

Element (E) is an RS-232 buffer (ADM222). The host system can communicate with the microcontroller through this RS-232 port.

3.2. Daughter board block diagram

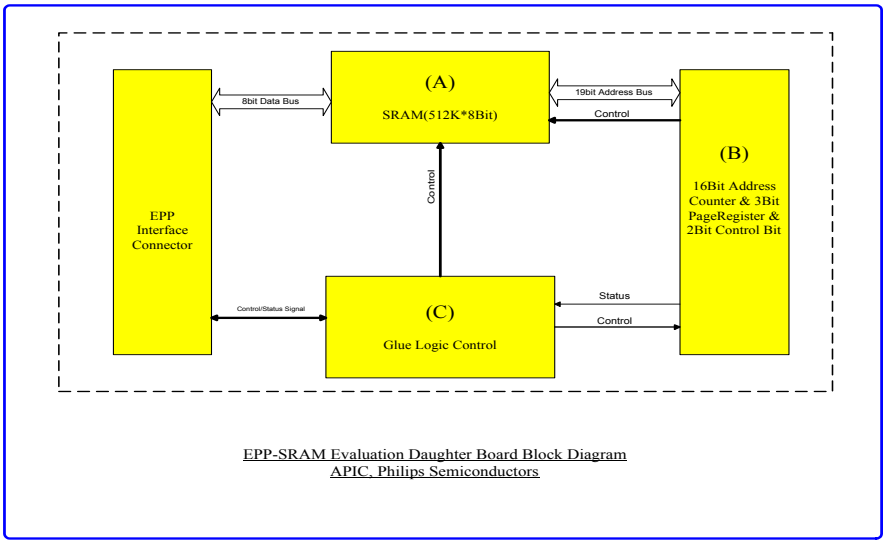

Figure 3-2: Daughter board block diagram

There are three elements on the daughter board. It contains the EPP interface for connecting to peripherals such as printer and scanner, and the RS-232 interface for connecting to the host system for firmware debugging.

Element (A) is an SRAM (512K * 8bit) for the main board to store data so that it can emulate the printer for receiving data (data pool) or the scanner for transmitting data (data source).

Element (B) includes a 14-bit counter, a 5-bit SRAM page switching, and a 2-bit control register.

Element (C) includes the decoding logic, which meets the EPP logic and timing requirement.

3.3. Description of the CPLD state machine

Figure 3-3 describes the DMA state machine of the CPLD. For DMA reading from D12 to the EPP interface, there are seven states for the state machine: IDLE/ DMAREAD1/ DMAREAD2/ DMAEND/ DMAEND1/ DMAEND2/ DMAEND3. For DMA reading from the EPP interface and writing to D12, there are seven states for the state machine: IDLE/ DMAWRIT1/ DMAWRIT2/ DMAEND / DMAEND1/ DMAEND2/ DMAEND3. The two states are different and the reset states can be shared by the DMA read or the DMA write.

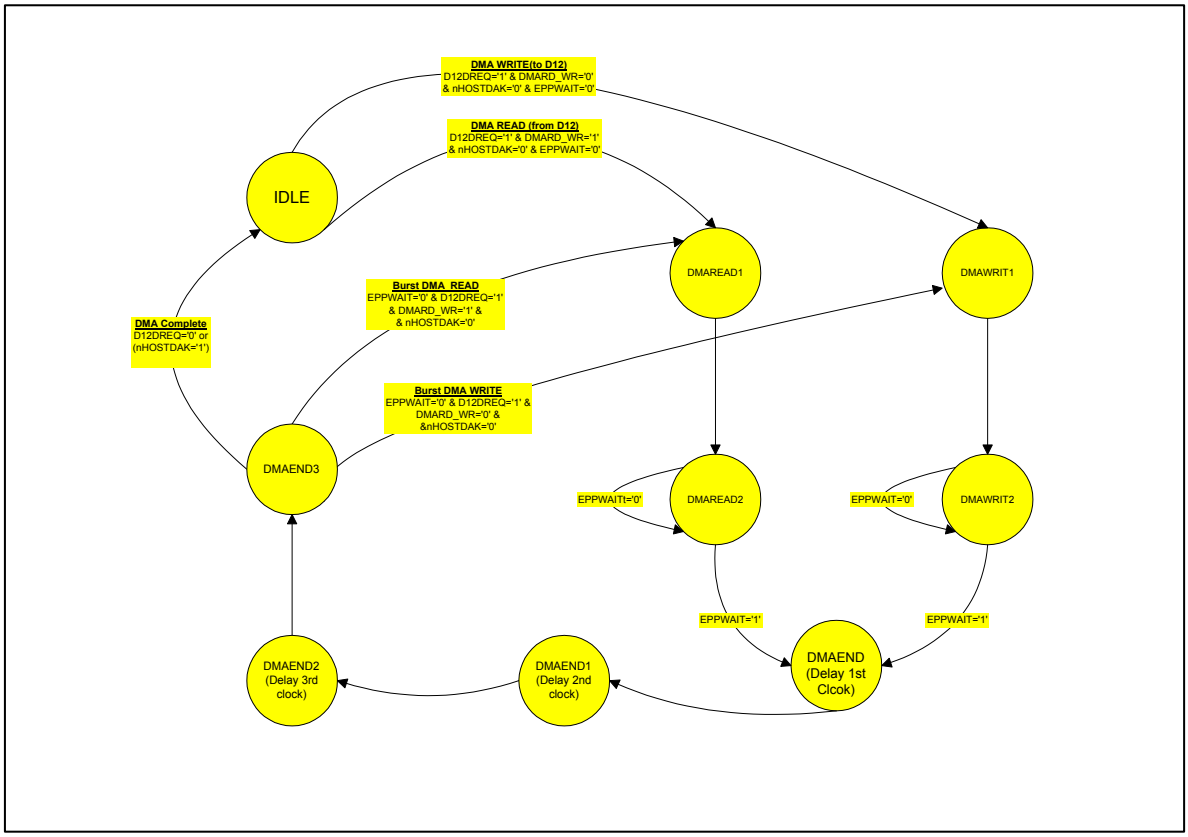

Figure 3-3: Flow diagram of the EPP and DMA controller (CPLD) state

3.4. Description of registers

3.4.1. Address mapping of CPLD internal registers and D12 registers

Table 3-1: Address mapping of CPLD and D12

MCU external memory address A1 and A0[1]	*Read/ Write*	*Register description D7–D0 (8-bit)*	*After reset (default value)*
00	R/W	DMA_BTS (7:0), counter LOW DMA burst transfer size, lower 8-bit	0x00
01	R/W	DMA_BTS (13:8) Counter HIGH DMA burst transfer size, upper 6-bit (Bits 7 and 6 are don't care)	0x00
10	R/W	PDIUSBD12 Data register (See D12 Data Spec.)	See Philips PDIUSBD12 spec.
11	R/W	PDIUSBD12 Command register (See D12 Data Spec.)	See Philips PDIUSBD12 spec.

[1] Latched from D1 and D0.

DMA_BTS is a 14-bit register, which can be accessed by the MCU (87C52). It can be divided into two registers, the lower 8-bit (address 0x0) and the upper 6-bit (address 0x1). When accessing this 14-bit register, you need to access it two times (addresses 0x0 and 0x1) because the data bus is only 8-bit wide.

This register (counter) is used in the USB device D12 DMA transfer mode. Before D12 starts the DMA transfer (DMA reads from D12 and writes to EPP, or DMA reads from EPP and writes to D12), the MCU needs to set the DMA burst transfer size (block size) to the CPLD (U5) DMA_BTS register.

During the DMA mode, the DMA_BTS register can be automatically decremented to one when one transaction (one read/write accessing) is complete. When this register reaches zero, it immediately generates a DMA end-of transfer (EOT) signal (active LOW) to the D12, stops the DMA request, then finishes the DMA transfer and clears the internal endpoint.

Note: Before DMA starts, the MCU needs to set DMARD_WR to 1 (DMA reads from the D12 mode) or 0 (DMA writes to the D12 mode). Otherwise, the DMA controller will get incorrect DMA direction transfer.

3.4.2. Description of the daughter board registers

3.4.2.1. Control and page selection register (CTRL_PAGE)

The CTRL_PAGE register is used to set the SRAM page number (ADR18-ADR14), clear ADDRCNTR, and preset the starting address for the SRAM. There are 32 pages for the SRAM page switching. Each page is 16 kbytes (ADR13-ADR0) in size, which is restricted by the CPLD internal DMA burst transfer size register. It has the same size as ADDRCNTR.

Bit CLEAR: After reset, the default value is 0. When the MCU writes 2-byte data to the Address register (/ASTROBE active), that is bit 7 (CLEAR) is set to one first and then reset to zero (positive pulse), the ADDRCNTR should be cleared to 0. When accessing the SRAM address from 0x00000, you need to perform the CLEAR operation.

Bit CNTRINC: After reset, the default value is zero. If you want ADDRCNTR to automatically increment to one after accessing the SRAM (read or write, /DSTROBE active), this bit should be set to zero. Otherwise, the ADDRCNTR output will stop incrementing. You can also use this bit to set the start of the SRAM address. First, use the CLEAR operation, write one and then zero to bit 6 (this bit) of the EPP Address register (/ASTROBE active). The ADDRCNTR should increment to one so that you can get the address you want by controlling loops by setting bit 6 to one and zero.

Bits ADR18-ADR14: After reset, the default value is B“00000”, which is the page number pointed to the SRAM (512K * 8 bit). The maximum number of pages is 32, that is, from 0 to 31.

Note: The CTRL_PAGE setting is controlled by /ASTROBE signal and not by /DSTROBE on the EPP interface.

Table 3-2: Description of CTRL_PAGE

Bit	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Function	CLEAR	CNTRINC	ADR18	ADR17	ADR16	ADR15	ADR14	X
Default	0	0	0	0	0	0	0	0

3.4.2.2. 14-bit address counter (ADDRCNTR)

The ADDRCNTR register is used as an address register on the daughter board that points to SRAM (512K * 8 bit), together with the page setting ADR18-ADR14 in the CTRL_PAGE register. After reset, the default value is 0x00000. This register can also be controlled by the CLEAR and CNTRINC registers.

Table 3-3: Description of ADDRCNTR

Bit	Bit13-Bit0	Default value
Function	ADR13-ADR0	B"00000000000000"

3.5. Signals description in the EPP mode

Table 3-4 summarizes the signals used for the EPP mode transfer. For detailed description, refer to clause 5 of the IEEE standard 1284 –1994.

Table 3-4: Signal description of EPP Mode

Compatibility mode signal name	EPP mode signal name	EPP mode signal description
nStrobe	nWrite	Set LOW to denote an address or data write operation to the peripheral. Set HIGH to denote an address or data read operation from the peripheral.
Data1 to Data 8	AD1 to AD8	Host–to-peripheral or peripheral-to-host address or data.
nAck	Intr	Used by the peripheral to interrupt the host. This signal is active high and positive edge triggered.
Busy	nWait	This signal should be driven inactive as a positive acknowledgement from the peripheral that transfer of data or address is completed. The signal is active LOW. It should be driven active to indicate that the device is ready for next address or data transfer.
nAutoFd	nDStrb	This signal is active LOW. It is used to denote a data cycle.
NInit	nInit	This signal is active LOW. When driven active (LOW), this signal initiates a termination cycle that results in the interface returning to compatibility mode.
nSelectIn	nAStrb	This signal is active LOW. It is used to denote an address cycle.
PError	User defined 1	This signal is manufacturer specific and beyond the scope of IEEE standard.
nFault	User defined 2	This signal is manufacturer specific and beyond the scope of IEEE standard.
Select	User defined 3	This signal is manufacturer specific and beyond the scope of IEEE standard.

Four operations supported on the EPP mode are:

- Address Write
- Data Write
- Address Read
- Data Read.

3.6. Miscellaneous

- Press-button S1 – reset or resume the system
- Press-button S2 – PIPE testing
- Press-button S3 – PIPE testing
- LED D1 – USB interface GoodLink indicator. When it is on, it means that the link between the USB host and the USB device has been set up and enumeration has been complete. If it is blinking, it means that data is being transmitted or received.
- LED D2 – PIPE testing
- LED D3 – PIPE testing
- +5 V power connector – DC 5 V input for the USB self-powered, internal pole is 5 V
- Power Supply Switch S4 – self-powered or bus-powered switch. If no DC 5 V input is provided, then it is used for power-off or bus-powered
- Ground header J11 – connected to ground for header testing

4. APPENDIX

4.1. Appendix A: Placement of the components of main and daughter board

Figure 4-1 and Figure 4-2 show the locations of the components on the main and daughter boards, the physical dimension, the drilling holes, and the pin layout of the IC. The dimensions of the main and daughter boards are 5.3” x 4.25” and 1.75” x 4.25” respectively. The total dimension of both boards is 7.05” x 4.25”.

The main components are as follows:

- U0: PAL16L8 programmable logic device
- U1: ADM222 CMOS RS-232 driver or receiver
- U2: microcontroller 87C52
- U3/U4: IEEE1284 interface buffer, 74HCT1284
- U5: CPLD (PZ5032)
- U6: USB device interface (PDIUSBD12)
- U7/U8: 74HCT393, dual 4-bit binary ripple counter
- U9: 512K x 8 bit SRAM, KM684000ALG

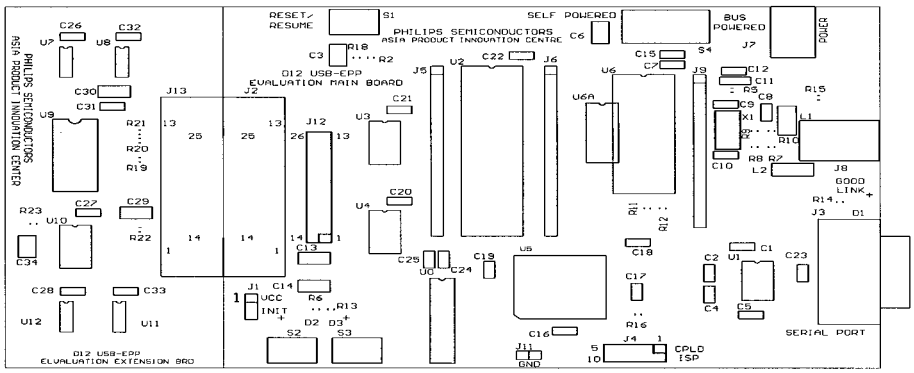

Figure 4-1: Top silkscreen layer of the main or daughter board

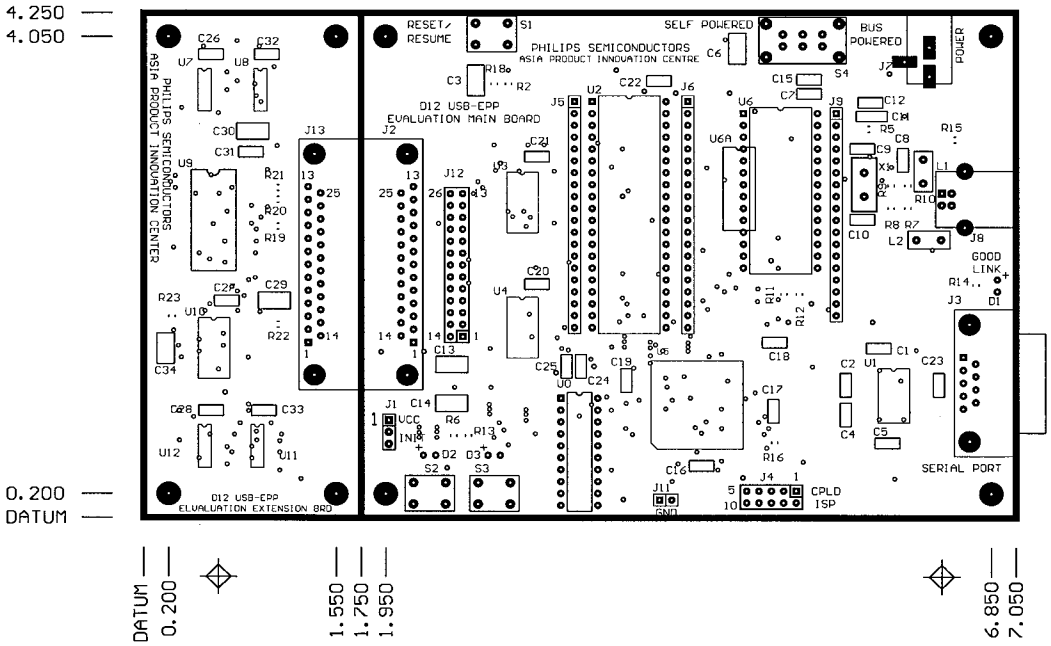

Figure 4-2: Top silkscreen overlays with top solder mask layer of the main or daughter board

4.2. Appendix B: Descriptions of the connectors and jumpers

The following tables provide detailed information on connectors and jumpers.

4.2.1. Microcontroller connectors J5 and J6

The expansion connector J5 is connected between pin 1 and pin 20 of the MCU. The expansion connector J6 is connected between pin 21 and pin 40 of the MCU. These connectors allow you to use MCU of other make through a pin converter by building a PCB of the same layout and pin definition as connectors J5 and J6. In addition, it allows you to debug signal lines from CPLD and PAL devices.

Table 4-1: Signal description of J5

Connector name	Pin no.	Signal name	Signal type	Description
J5	1 (square pin)	/WRITE	I/O	EPP write, active LOW
J5	2	/DSTROBE	I/O	EPP data strobe, active LOW
J5	3	/ASTROBE	I/O	EPP address strobe, active LOW
J5	4	WAIT	I	EPP wait, active HIGH
J5	5	/INTR	I	EPP interrupt, active LOW
J5	6	PE	I	Paper end
J5	7	/ERROR	I	ERROR, active LOW
J5	8	SELECT	I	Select device
J5	9	MCURST	I	MCU reset, active HIGH
J5	10	S232RXD	I	RS-232 RXD
J5	11	S232TXD	O	RS-232 TXD
J5	12	/D12INT	I	D12 interrupt, active LOW
J5	13	SUSPEND	I	Suspend mode
J5	14	RLED0	O	Red LED 0
J5	15	RLED1	O	Red LED 1
J5	16	/MCUWR	I/O	MCU write to external memory, active LOW
J5	17	/MCURD	I/O	MCU read from external memory, active LOW
J5	18	MCUX2	O	Crystal
J5	19	CLK12M	O	Clock output
J5	20	GND	Power	Ground

Table 4-2: Signal description of J6

Connector name	Pin no.	Signal name	Signal type	Description
J6	1 (square pin)	V_{CC}	Power	+5 V
J6	2	D0	I/O	Microcontroller data bus
J6	3	D1	I/O	Microcontroller data bus
J6	4	D2	I/O	Microcontroller data bus
J6	5	D3	I/O	Microcontroller data bus
J6	6	D4	I/O	Microcontroller data bus
J6	7	D5	I/O	Microcontroller data bus
J6	8	D6	I/O	Microcontroller data bus
J6	9	D7	I/O	Microcontroller data bus
J6	10	V_{CC}	Power	+5 V
J6	11	ALE	O	Address latch enable, memory mode
J6	12	/PSEN	I	Program store enable, active LOW means external memory
J6	13	ALE1	O	Address latch enable, I/O mode
J6	14	SWM0	I	Switch mode 0
J6	15	SWM1	I	Switch mode 1
J6	16	DMARD_WR	O	DMA transfer direction, read from D12/write to D12
J6	17	WH ICH_PW	I	Bus power (HIGH) or self-power (LOW)
J6	18	/EPPINIT	O	EPP initialization
J6	19	/HOSTDACK	O	Host DMA acknowledgment
J6	20	HOSTDREQ	I	Host DMA request

4.2.2. Test header J9

The J9 test header for the D12 USB interface device is used to detect or probe D12 signals other than the data bus. The data bus, however, can be detected from J6. Some signals of CPLD are also connected to this test header.

Table 4-3: Signal description of J9

Connector name	Pin no	Signal name	Signal type	Description
J9	1(square pin)	D12A0	I	Address A0
J9	2	+3.3 V	Power	3.3 V power supply
J9	3	D12DP	I/O	USB data plus line
J9	4	D12DM	I/O	USN data minus line
J9	5	V_{cc}	Power	+5 V
J9	6	XTAL2	O	Crystal
J9	7	XTAL1	I	Crystal
J9	8	GOODLNK	O	GoodLink for the USB interface data link status
J9	9	V_{cc}	Power	+5 V
J9	10	/D12EOT	I	End of DMA transfer
J9	11	/D12DACK	I	DMA acknowledge
J9	12	D12DREQ	O	DMA request
J9	13	/D12WR	I	IO write to D12
J9	14	/D12RD	I	IO read to D12
J9	15	/D12INT	O	D12 interrupt out
J9	16	CLK12M	O	D12 clock output for other device
J9	17	SUSPEND	I/O	Suspend output or input
J9	18	/D12CS	I	D12 chip select

4.2.3. ISP Connector J4

This 10-pin connector is used for CPLD In-System Programming (ISP) if PZ5032CS10A44 is used (ISP version). It is not useful for PZ5032-7A44 (Non-ISP version). If you do not have a CPLD programmer, you only need to use this connector for directly programming U5 CPLD. Philips provides the programming cable, which connects J4 to the PC parallel port. More information can be obtained from the Website: http://www.coolcpld.com/.

Table 4-4: Signal description of J4

Connector name	Pin no.	Signal name	Signal type	Description
J4	1(square pin)	GND	Power	Ground
J4	2	GND	Power	Ground
J4	3	NC	Not connected	
J4	4	GND	Power	Ground
J4	5	NC	Not connected	
J4	6	CPLDTCK	I	JTAG, TCK
J4	7	CPLDTMS	I	JTAG, TMS
J4	8	CPLDTDI	I	JTAG, TDI
J4	9	CPLDTDO	O	JTAG, TDO
J4	10	NC	Not connected	

4.2.4. USB serials-B connector J8 (upstream)

The J8 connector is a standard USB serial B connector that has two data lines, V-Bus and Ground. The connector can be directly connected to the system or any USB hub downstream port. More details can be found in the USB specification.

Table 4-5: Signal description of J8

Connector name	Pin no.	Signal name	Signal type	Description
J8	1 (square pin)	V_BUS	Power	+5 V
J8	2	D-	I/O	Data minus line
J8	3	D+	I/O	Data plus line
J8	4	GND	Power	Ground

4.2.5. RS-232 interface connector J3

The RS-232 connector is used for debugging. It can be connected to the PC for monitoring the status of the D12 device. Only two signals are used, that is, signal to receive data and signal to transmit data. The debugging feature is such that the MCU will print all activities related to the USB specification to any PC terminal. This feature allows developers to trap any failure, especially during the enumeration process.

Table 4-6: Signal description of J3

Connector name	Pin no.	Signal name	Signal type	Description
J3	1(square pin)	NC	No connected	
J3	2	RS-232TXD	O	RS-232 transmit data line
J3	3	RS-232RXD	I	RS-232 receive data line
J3	4	NC	Not connected	
J3	5	GND	Power	Ground
J3	6	NC	Not connected	
J3	7	NC	Not connected	
J3	7	NC	Not connected	
J3	9	NC	Not connected	

4.2.6. Jumper J1

The jumper J1 is for supplying power (5 V) to the daughter board and to allow you to connect to other EPP devices.

Table 4-7: Pin configurations of J1

Jumper name (J1)	Pins connections	Description
CASE 1	Pin 2 connected to pin 1 (shorted together by the jumper block)	Pin 1of jumper J1 is V_{CC}. Pin 2 is connected to pin 16 (/INIT) of connector J2 or pin 16 of connector J13. For the daughter board, pin 16 of J3 must be 5 V. In this case (for connecting the daughter board to the main board), pin 2 and pin 1 must be enabled.
CASE 2	Pin 2 connected to pin 3 (short together by the jumper block)	Pin 2 of jumper J1 is connected to pin 16 of connector J2, and J3 is connected to pin 14 of U4 (/PINIT). If the main board needs to be connected to the EPP interface, pin 2 and pin 3 of J1must be enabled.

4.2.7. Parallel Port connector and its test header

The signal for the connectors J2 (female-type on the main board), J13 (male-type on the daughter board), and J12 (the test header) is fully compliant with the IEEE standard 1284–1994. J2 and J13 are D-type 25-pin female/male connectors while J12 is a 26-pin header for debugging.

Table 4-8: Signal description of J2, J13 and J12

Connector name	Test header name	Pin no.	Signal name	Signal type	Description
J2/J13	J12	1	/STROBE	O	Strobe D7-D0
J2/J13	J12	2	PD0	I/O	Data bus
J2/J13	J12	3	PD1	I/O	Data bus
J2/J13	J12	4	PD2	I/O	Data bus
J2/J13	J12	5	PD3	I/O	Data bus
J2/J13	J12	6	PD4	I/O	Data bus
J2/J13	J12	7	PD5	I/O	Data bus
J2/J13	J12	8	PD6	I/O	Data bus
J2/J13	J12	9	PD7	I/O	Data bus
J2/J13	J12	10	/ACK	I	Acknowledgment, may trigger interrupt
J2/J13	J12	11	BUSY	I	Busy, printer busy
J2/J13	J12	12	PE	I	Paper end, empty (out of paper)
J2/J13	J12	13	SELECT	I	Printer selected (on-line)
J2/J13	J12	14	/AUTOFD	O	Generate automatic line feeds after carriage returns
J2/J13	J12	15	/ERROR	I	Error
J2/J13	J12	16	/INIT	O	Initialize
J2/J13	J12	17	/SELECTIN	O	Select device
J2/J13	J12	18	GND for J2, NC for J12	Power	Ground
J2/J13	J12	19	GND for J2, NC for J12	Power	Ground
J2/J13	J12	20	GND for J2, NC for J12	Power	Ground
J2/J13	J12	21	GND for J2, NC for J12	Power	Ground
J2/J13	J12	22	GND for J2, NC for J12	Power	Ground
J2/J13	J12	23	GND for J2, NC for J12	Power	Ground
J2/J13	J12	24	GND for J2, NC for J12	Power	Ground
J2/J13	J12	25	GND for J2, NC for J12	Power	Ground
	J12	26	GND	Power	Ground

4.2.8. IEEE1284—A connector description

IEEE 1284 defines five communication modes. It uses the terms FORWARD CHANNEL to refer to the transfer from the host to the peripherals and REVERSE CHANNEL to refer to the transfer from the peripheral to the host. The five modes are Compatibility, Nibble, Byte, EPP, and ECP.

With so many modes to choose from, the host and peripheral need to decide which mode they intend to use to communicate with one another. IEEE 1284 negotiation phase enables devices to talk back and forth so that they can choose the best mode. Through negotiation, the host can find out which mode would be supported by the peripheral. If a peripheral supports multiple modes, the negotiation will inform the peripheral which mode the host wishes to use.

For the purpose of this eval kit, the Enhanced Parallel Port (EPP) mode is preselected so that data is transferred at a higher speed in both directions. EPP can distinguish between two types of information, which is usually defined as data and addresses. In the EPP mode, there are only two types of information: data and address transfer.

Table 4-9 shows the pin numbers and their assigned signal names for the IEEE1284–A connectors. The 1284-A connector is a 25-pin subminiature D-shell connector.

Table 4-9: Signal description of IEEE 1284-A

Pin no	Source	Compatibility mode	Nibble mode	Byte mode	EPP mode	ECP mode
1	H	nStrobe	HostClk	HostClk	nWrite	HostClk
2	Bi-Dir	Data Bit1 (LSB)	Data Bit1 (LSB)	Data Bit1 (LSB)	Address/Data Bit1	Data Bit1 (LSB)
3	Bi-Dir	Data Bit2	Data Bit2	Data Bit2	Address/Data Bit2	Data Bit2
4	Bi-Dir	Data Bit3	Data Bit3	Data Bit3	Address/Data Bit3	Data Bit3
5	Bi-Dir	Data Bit4	Data Bit4	Data Bit4	Address/Data Bit4	Data Bit4
6	Bi-Dir	Data Bit5	Data Bit5	Data Bit5	Address/Data Bit5	Data Bit5
7	Bi-Dir	Data Bit6	Data Bit6	Data Bit6	Address/Data Bit6	Data Bit6
8	Bi-Dir	Data Bit7	Data Bit7	Data Bit7	Address/Data Bit7	Data Bit7
9	Bi-Dir	Data Bit8 (MSB)	Data Bit8 (MSB)	Data Bit8 (MSB)	Address/Data Bit8	Data Bit8 (MSB)
10	P	nAck	PtrClk	PtrClk	Intr	PeriphClk
11	P	Busy	PtrBusy	PtrBusy,	nWait	PeriphAck
12	P	PaperEnd	AckDataReq	AckDataReq,	User Defined Bit 1	nAckReverse
13	P	Select	XFlag	XFlag,	User Defined Bit 3	Xflag
14	H	nAutoLF	HostBusy	HostBusy	nDstrb	HostAck
15	P	nFault	nDataAvail	nDataAvail	User Defined Bit 2	nPeriphReq
16	H	nInit	nInit	nInit	nInit	nReverseReq
17	H	nSelectIn	1284Active	1284Active	nAstrb	1284Active
18		Signal Ground	Signal Ground	Signal Ground	Signal Ground	Signal Ground
19		Signal Ground	Signal Ground	Signal Ground	Signal Ground	Signal Ground
20		Signal Ground	Signal Ground	Signal Ground	Signal Ground	Signal Ground
21		Signal Ground	Signal Ground	Signal Ground	Signal Ground	Signal Ground
22		Signal Ground	Signal Ground	Signal Ground	Signal Ground	Signal Ground
23		Signal Ground	Signal Ground	Signal Ground	Signal Ground	Signal Ground
24		Signal Ground	Signal Ground	Signal Ground	Signal Ground	Signal Ground
25		Signal Ground	Signal Ground	Signal Ground	Signal Ground	Signal Ground

4.3. Appendix C: Main and daughter board schematics

4.3.1. Main board schematic

The schematic for the main board, which is drawn on the OrCAD V7.0 EDA platform, is shown in Figure 4-3.

4.3.2. Daughter board schematic

The schematic for the daughter board, which is drawn on the OrCAD V7.0 EDA platform, is shown in Figure 4-4.

]

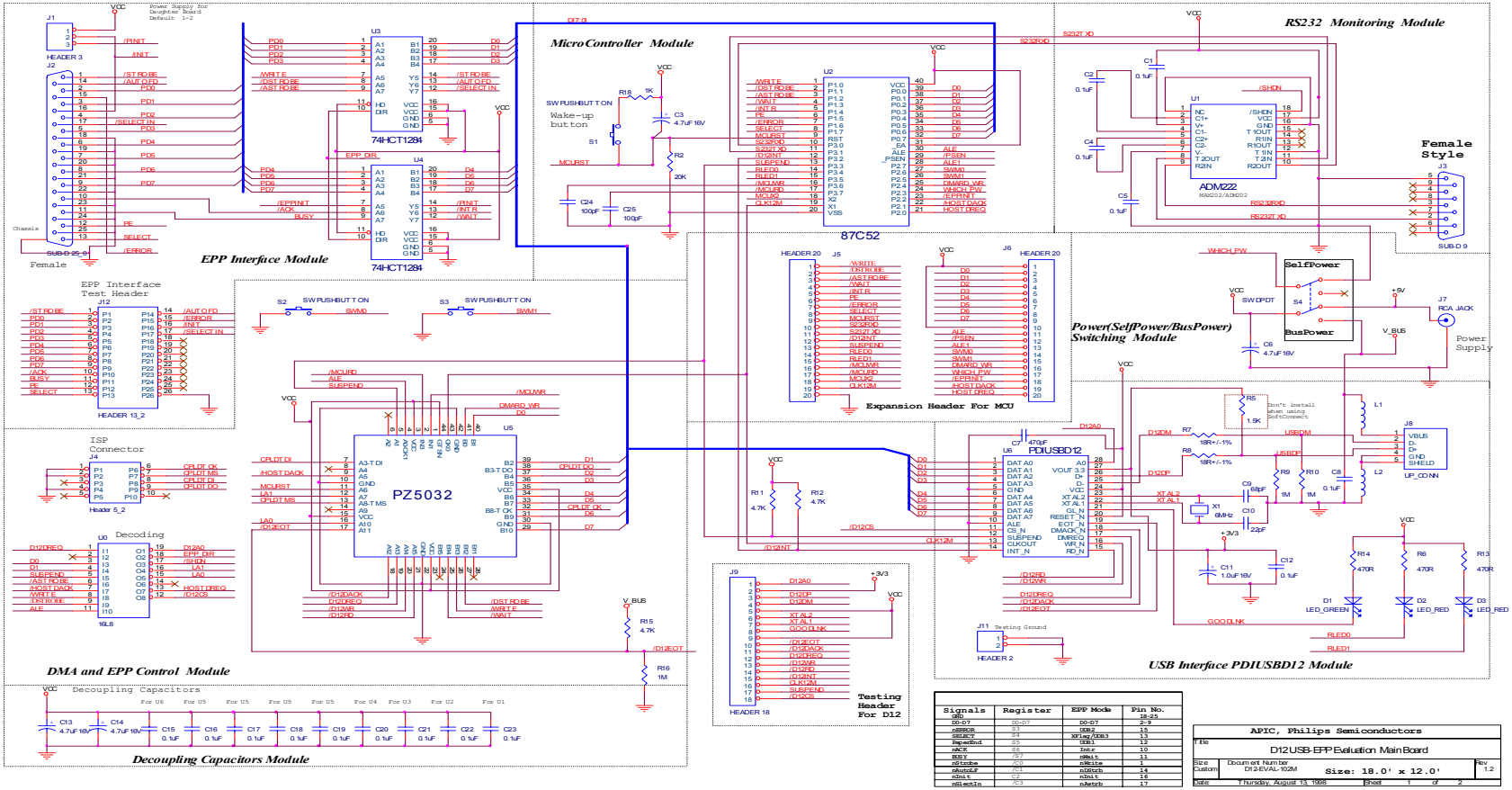

Figure 4-3: Schematic of the main board

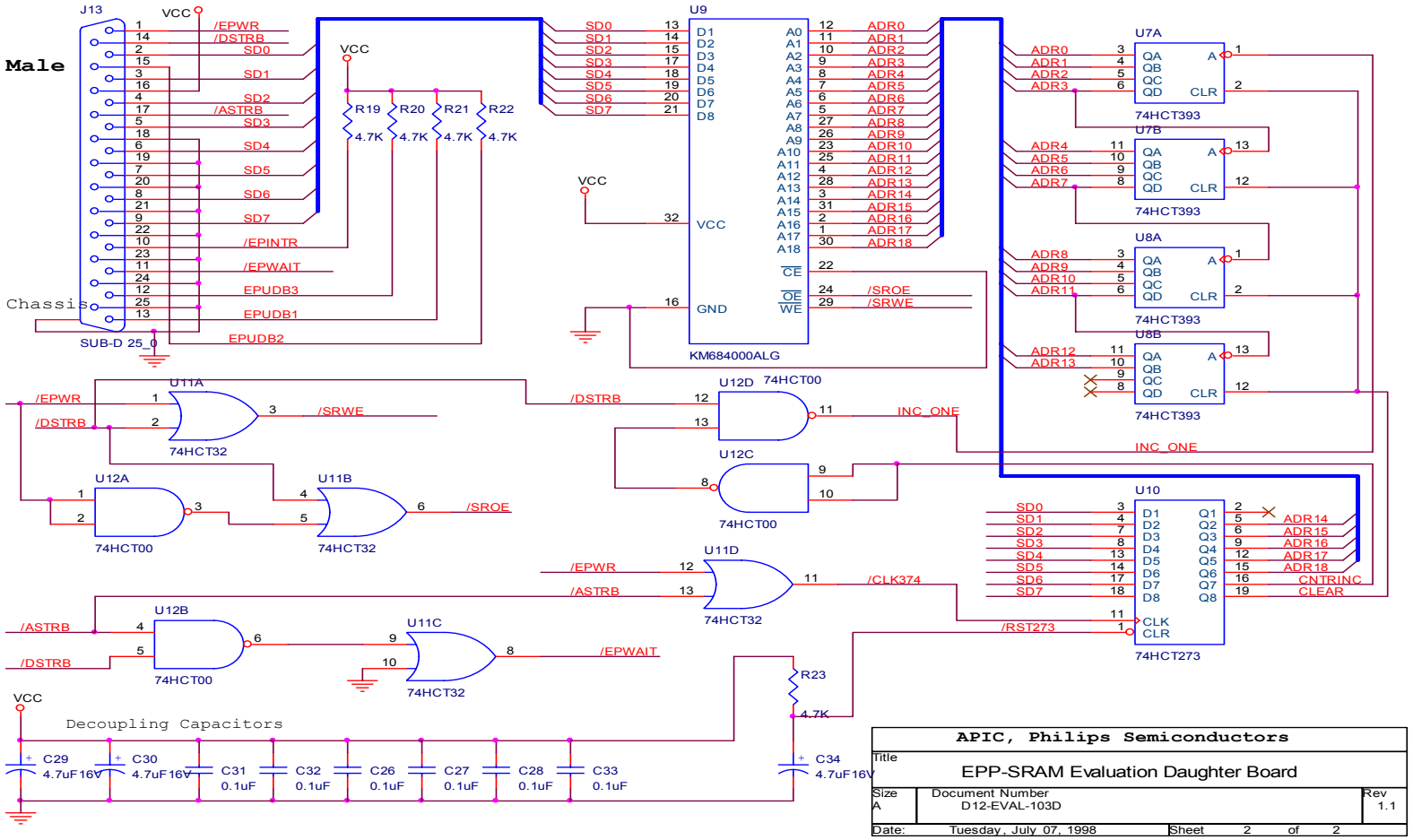

Figure 4-4: Schematic of the daughter board

4.4. Appendix D: Bill of materials (BOM)

4.4.1. Main board BOM

Table 4-10: BOM for main board

No.	Item P/N and description	Reference	Qty	Package	Approved vendors	Remark
M.1	PDIUSBD12, Philips USB Interface device with parallel interface	U6	1	DIP28 (600 mil) and SO28	Philips Semiconductors	
M.2	PZ5032CSA44, Philips 32 Micro-Cell CPLD	U5	1	PLCC44	Philips Semiconductors	
M.3	P87C52UBPN, Philips 8051 compatible Microcontroller	U2	1	DIP40, 600 mil	Philips Semiconductors	
M.4	74HCT1284D, Philips IEEE1284 buffer	U3, U4	2	SOL20, 20-pin plastic	Philips Semiconductors	
M.5	ADM222/ADM202, CMOS RS-232 Driver/Receiver	U1	1	SOIC16 (ADM202)/SOIC18 (ADM222)	Analog Devices	
M.6	16L8-15,PAL device	U0	1	DIP20, 300 mil	Open Source	
M.7	6.0 MHz Crystal, TTL or CMOS Output with 0.01% Frequency Tolerance, 40%-60% Duty Cycle	X1	1	Thru' hole, Low Profile, Lead's Pitch = 0.3	Open Source	
M.8	Push-Button, Normally open	S1, S2, S3	3	Thru' hole	Open Source	
M.9	Double Pole and Double Thread Switch, DPDT	S4	1	Thru' Hole	Open Source	
M.10	Green Color LED	D1	1	Thru' Hole, Lead's Pitch = 0.1	Open Source	
M.11	Red Color LED	D2, D3	2	Thru' Hole, Lead's Pitch = 0.1	Open Source	
M.12	DB 9-pin/Female D-Type/Right angle PCB Mount Connector	J3	1	Thru' Hole	Open Source	
M.13	DB 25pin/ Female D-Type/Right Angle PCB Mount Connector	J2	1	Thru' Hole	Open Source	
M.14	RCA Jack for Self Power supply (+5 V)	J7	1	Thru' Hole	Open Source	
M.15	USB B-Type Connector (Upstream)	J8	1	Thru' Hole	Open Source	
M.16	Single Row, 2-pin Straight Header, Solder Type	J11	1	SIL2, Lead's Pitch = 0.1	Open Source	
M.17	Single Row, 3-pin Straight Header, Solder Type	J1	1	SIL3, Lead's Pitch = 0.1	Open Source	
M.18	Single Row, 18-pin Straight Header, Solder Type	J9	1	SIL18, Lead's Pitch = 0.1	Open Source	
M.19	Single Row, 20-pin Straight Header, Solder Type	J5, J6	2	SIL20, Lead's Pitch = 0.1	Open Source	

No.	Item P/N and description	Reference	Qty	Package	Approved vendors	Remark
M.20	Double Row, 10-pin Straight Header, Solder Type	J4	1	DIL10, Lead's Pitch = 0.1	Open Source	
M.21	Double Row, 26-pin Straight Header, Solder Type	J12	1	DIL26, Lead's Pitch = 0.1	Open Source	
M.22	Ferrite Bead, Multi-layer Chip Bead (10 nH)	L1, L2	2	1206	Open Source	
M.23	18 R resistor, 1/10W, +/-1%	R7, R8	2	0805	Open Source	
M.24	470 R resistor, 1/10W, +/-5%	R6, R13, R14	3	0805	Open Source	
M.25	1.0 K resistor, 1/10W, +/-5%	R18	1	0805	Open Source	
M.26	1.5 K resistor, 1/10W, +/-5%	R5	1	0805	Open Source	
M.27	4.7 K resistor, 1/10W, +/-5%	R11, R12, R15	3	0805	Open Source	
M.28	20 K resistor, 1/10W, +/-5%	R2	1	0805	Open Source	
M.29	1 M resistor, 1/10W, +/-5%	R9, R10, R16	3	0805	Open Source	
M.30	22 pF Capacitor, 50V, Multilayer, +/-20%	C10	1	0805	Open Source	
M.31	68 pF Capacitor, 50V,Multilayer, +/-20%	C9	1	0805	Open Source	
M.32	100 pF Capacitor, 50V, Multi-layer, +/-20%	C24, C25	2	0805	Open Source	
M.33	470 pF Capacitor, 50V, Multi-layer, +/-20%	C7	1	0805	Open Source	
M.34	0.1 uF Capacitor, 50V,Multi-layer, +/-20%	C1, C2, C4, C5, C8, C12, C15, C16, C17, C18, C19, C20, C21, C22, C23, C24,	16	0805	Open Source	
M.35	1.0 uF Capacitor, 16V, Electrolytic/multi-layer Chip Bead, +/-20%	C11	1	Case "C"	Open Source	
M.36	4.7 uF Capacitor, 16V, +/-20%, Electrolytic/Multilayer Chip Bead	C3, C6, C13, C14,	4	Case "C"	Open Source	
M.37	2-Layer PCB, USB-EPP Main Board, Size 5.3" * 4.25"	2 Layers	1	Rectangle	ESA	

4.4.2. Daughter Board BOM

Table 4-11: BOM for daughter board

No.	Item P/N and description	Reference	Qty	Package	Approved vendors	Remark
D.1	74HCT393, CMOS Dual 4-Bit Ripple Counter	U7, U8	2	SOIC14	Open Source	
D.2	KM6840000ALG, 512K * 8Bit SRAM	U9	1	32-Pin SOP	Samsung	
D.3	74HCT273, Octal D-Type Flip-Flop With Reset	U10	1	SOIC20	Open Source	
D.4	74HCT32,Quad 2-Input OR Gate	U11	1	SOIC14	Open Source	
D.5	74HCT00,Quad 2-Input NAND Gate	U12	1	SOIC14	Open Source	
D.6	DB 25pin Male D-Type/Right angle Connector	J13	1		Open Source	
D.7	4.7 K resistor, 1/10W, +/- 5%	R19, R20, R21, R22, R23	1	0805	Open Source	
D.8	0.1 uF Capacitor, 50V, Multilayer, +/- 20%	C26, C27, C28, C31, C32, C33	1	0805	Open Source	
D.9	4.7 uF Capacitor, 16V, Electrolytic or Multilayer Chip bead +/- 20%	C29, C30, C34	3	Case “C”	Open Source	
D.10	Two-layer PCB, USB-EPP Daughter Board, Size 1.75” * 4.25”	2 Layers	1	Rectangle	ESA	

4.5. Appendix E: CPLD VHDL simulation waveforms

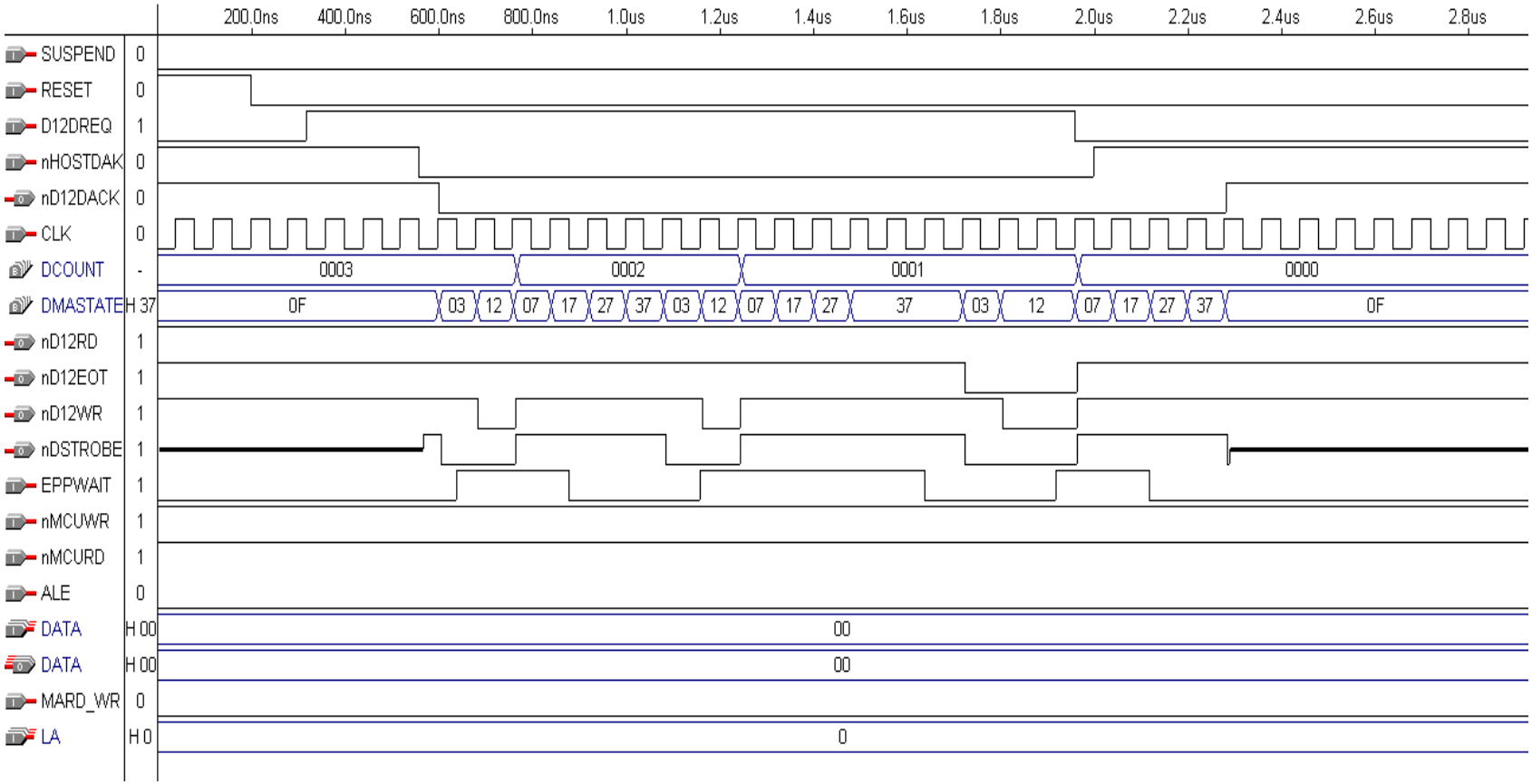

Figure 4-5: DMA write timing (Reading from EPP and writing to D12)

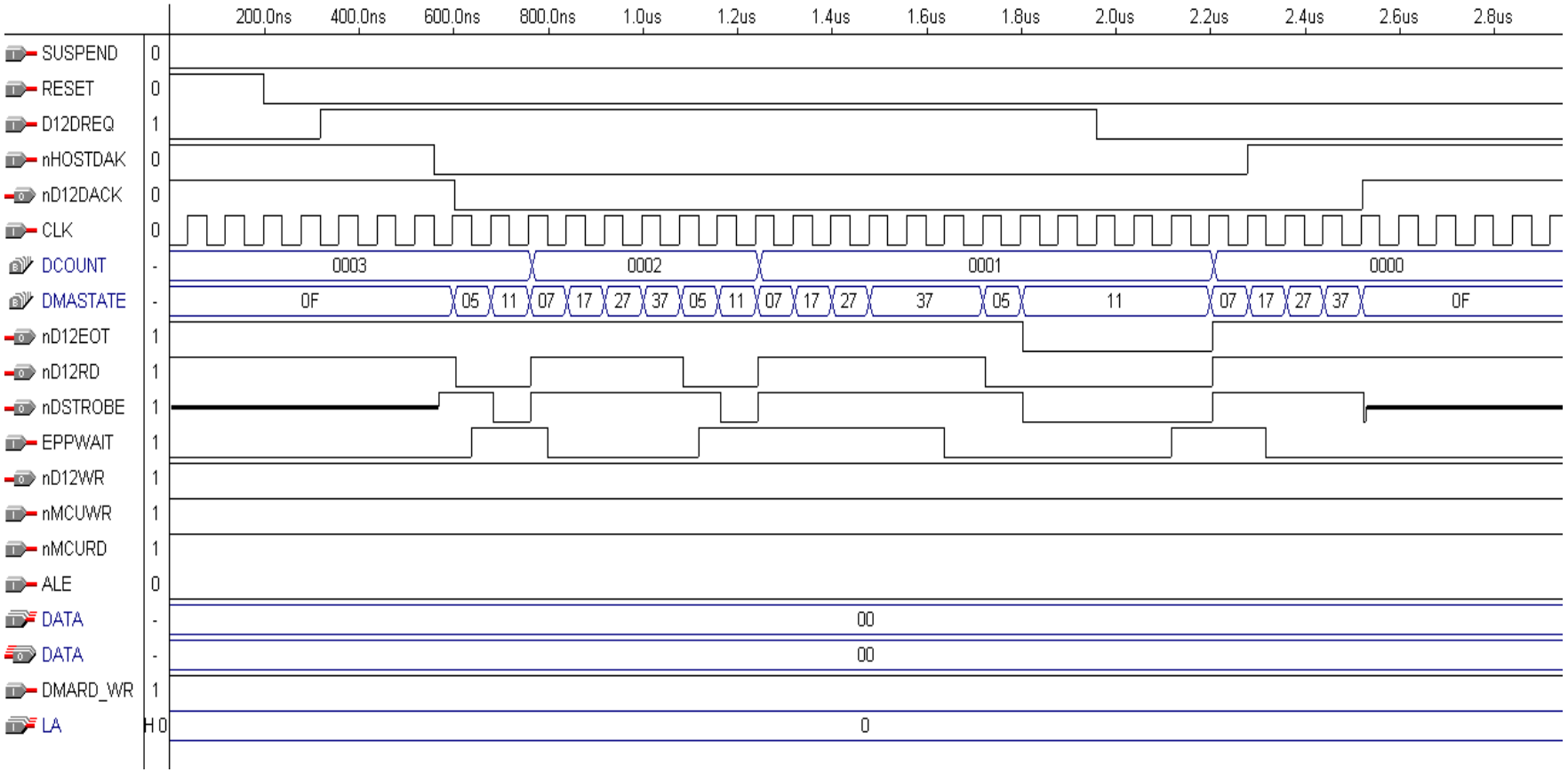

Figure 4-6: DMA read timing (Reading from D12 and writing to EPP)

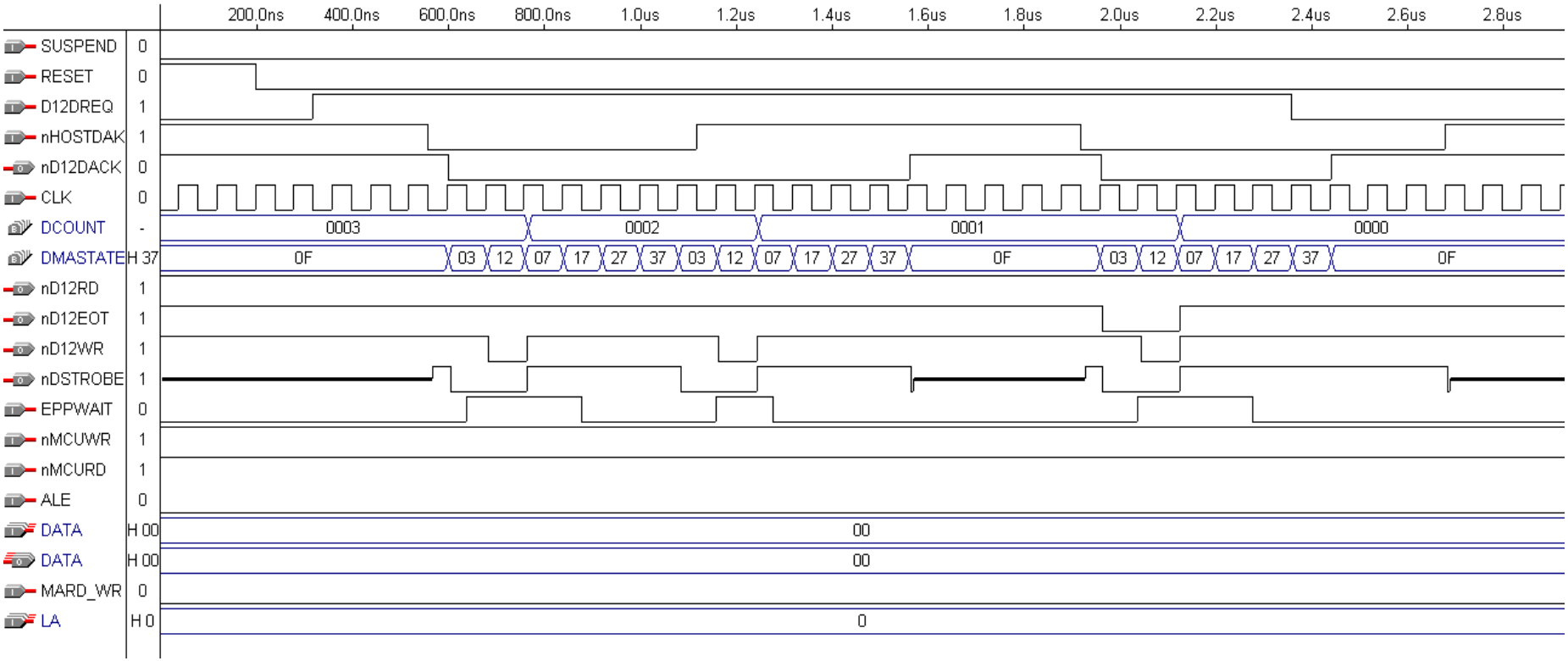

Figure 4-7: DMA read timing (Reading from D12 and writing to EPP) with nHOSTDAK inactive during DMA transfer

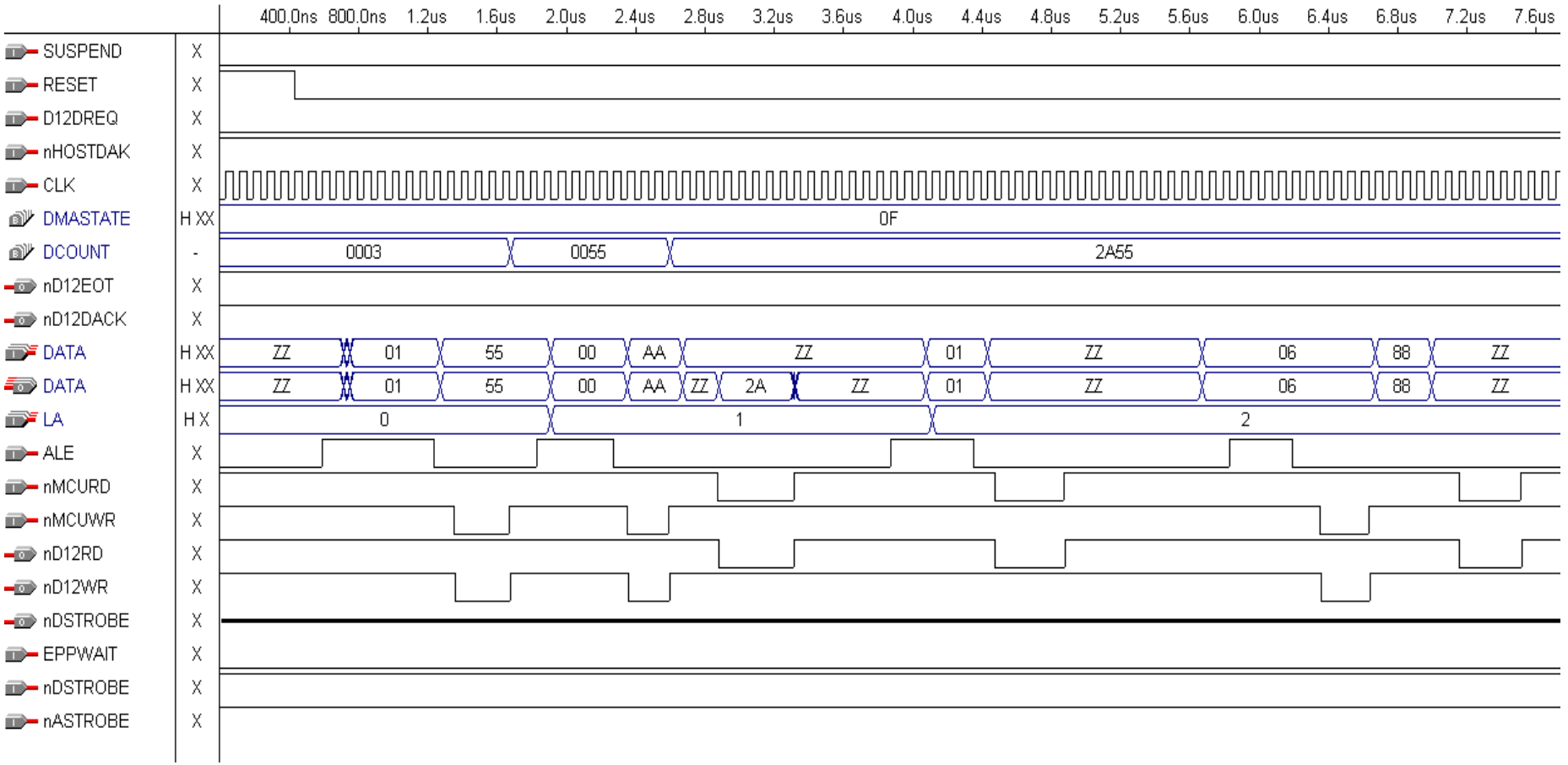

Figure 4-8: MCU accessing CPLD and D12 timing

4.6. Appendix F: CPLD VHDL source codes

```
--*************************************************************************
--
--    Project Name :    USB-EPP Eval Kit
--  Chip(CPLD)Name: DMA-EPP Controller
--  VHDL Code:        EPPDMA.VHD (simplified version for State Machine)
--  Target CPLD: PZ5032cs10A44
--    Designer:           Steven Cheng
--    Copy Right:  APIC, Philips (Semiconductors) Singapore Pte Ltd
--    Last updated:Sept. 28 , 1998
--  Version;          1.04
--*************************************************************************
--
--
--
--
--
--
--
--*************************************************************************

library ieee; -- IEEE general Library
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
--library ASYL; --Philips CPLD library
--use ASYL.SL_ARITH.all;
--use ASYL.ARITH.all;
-------------------------------------------------------------------------
entity EPPDMA is
   port(
     CLK :          in         std_logic; --comman Clock 12Mhz
     RESET:         in         std_logic; -- high active
     DATA:          inout      std_logic_vector(7 downto 0); -- data bus
     LA:            in         std_logic_vector(1 downto 0);--latched address bus
     DMARD_WR:      in         std_logic; -- DMA transfer direction, 1=read from
D12


     ---D12 Signals ----------
     D12DREQ:   in    std_logic; -- D12 DMA request
     nD12WR:          buffer    std_logic; -- D12 Write
     nD12RD:          buffer    std_logic; -- D12 Read
     nD12DACK: buffer    std_logic; -- D12 DMA ACK
     nD12EOT:   out  std_logic; -- D12 End of Transfer
     --SUSPEND: in    std_logic; -- Suspend mode

     ----MCU Signals ----------
     nMCUWR:          in    std_logic; -- MCU write
     nMCURD:          in    std_logic; -- MCU read
     --ALE:           in    std_logic; -- Address latch Enable
     nHOSTDAK:        in    std_logic; -- Host DMA ACK

     ----EPP Signals
     EPPWAIT:   in    std_logic; -- EPP Wait
     --RWCLK:          buffer    std_logic;
     --RWCLKIN:In          std_logic;

     nDSTROBE: out  std_logic -- EPP Data Strobe
     --nEPPWR:          out       std_logic; -- EPP Write,
     );

end EPPDMA;
-------------------------------------------------------------------------------

architecture behave of EPPDMA is
-- LA="00" ->  CPLD internal Register Low;
-- LA="01" ->  CPLD Internal Register High;
-- LA="10" ->  D12 Data Register;
```

```
-- LA="11" ->  D12 Command Register;

--------State Machine Definition
      --Trueth Table
      -- State        DELAY1 DELAY0 nD12DACK nDSTROB        nD12RD
      nD12WR
      -- IDLE          0       0       1       1           1           1
      -- DMAREAD1      0       0       0       1           0           1
      -- DMAREAD2      0       1       0       0           0           1


      -- DMAWRIT1      0       0       0       0           1           1
      -- MMAWRIT2      0       1       0       0           1           0

      -- DMAEND        0       0       0       1           1           1
      -- DMAEND1           0       1       0       1           1           1
      -- DMAEND2           1       0       0       1           1           1
      -- DMAEND3           1       1       0       1           1           1


      ----DCOUNT Parameters
      Constant BITW:        integer:= 13; -- Width of CPLD internal Counter-1
      Constant BITWZERO:    std_logic_vector(15 downto (BITW+1)):="00"; --
      Constant BITWZERO1:   std_logic_vector(BITW downto
0):="00000000000011"; --
      Constant BITWZERO2:   std_logic_vector(BITW downto
1):="0000000000000"; --

      signal DOUT:          std_logic_vector (7 downto 0); --Data output
      --signal DELAY:       integer range  0 to 3; --delay cycle for D12 READ High

    signal DIN:           std_logic_vector (7 downto 0); --Data Input
    signal DCOUNT:        std_logic_vector (BITW downto 0); --CPLD Internal
Counter

      signal DMA_ON:        std_logic; -- DMA is on Cycle --Aug 8
      signal RWCLK:         std_logic; -- Clock for internal Counter
      signal DCZERO:        std_logic; -- Counter return to Zero
      signal nDSTROB:       std_logic; -- EPP Data Strobe


      signal DMASTATE:      std_logic_vector(5 downto 0); -- State Machine

      Constant IDLE:        std_logic_vector(5 downto 0):="001111"; --IDLE state
      Constant DMAREAD1:    std_logic_vector(5 downto 0):="000101"; -- DMA
read state 1
      Constant DMAREAD2:    std_logic_vector(5 downto 0):="010001"; -- DMA
read State 2

      Constant DMAWRIT1:    std_logic_vector(5 downto 0):="000011"; -- DMA
write State 1
      Constant DMAWRIT2:    std_logic_vector(5 downto 0):="010010"; -- DMA
write State 2

      Constant DMAEND:  std_logic_vector(5 downto 0):="000111"; -- Current DMA
R/W End
      Constant DMAEND1:std_logic_vector(5 downto 0):="010111"; -- Current DMA
R/W End
      Constant DMAEND2:std_logic_vector(5 downto 0):="100111"; -- Current DMA
R/W End
      Constant DMAEND3:std_logic_vector(5 downto 0):="110111"; -- Current DMA
R/W End


-------------------------------------------------------------------------------


begin

---*************************************************************************
STATE: process (RESET,CLK)
begin
```

```
if RESET='1' then
        DMASTATE <=IDLE;

elsif CLK'event and CLK='1' then
-------------------------------------------------
    --- State Transition -----
  Case  DMASTATE  is

        ------IDLE state --------
      when IDLE =>
         if nHOSTDAK='0' and EPPWAIT='0'  then
              --if D12DREQ='1' and (DCZERO='0' or DCOUNT(0)='1')
then
              if D12DREQ='1'  then
                   if DMARD_WR='1' then
                        DMASTATE <=DMAREAD1; --- DMA Read Transfer

                   else
                        DMASTATE <=DMAWRIT1; -- DMA write Transfer

                   end if;
              end if;
         end if;


         ---DMA read Cycle----------------------
         when DMAREAD1 =>
              --if  EPPWAIT='1' then
                   DMASTATE <= DMAREAD2; -- Read
              --end if;


         when DMAREAD2 =>
              if  EPPWAIT='1' then
                   DMASTATE <= DMAEND; -- Current DMA transfer finished
              else
                   DMASTATE <=DMAREAD2;
              end if;
         ----------------------------------------


         ---DMA read Cycle----------------------
         when DMAWRIT1 =>
              DMASTATE <= DMAWRIT2; -- write

         when DMAWRIT2 =>
              if EPPWAIT='1' then
                   DMASTATE <= DMAEND; -- Current DMA transfer finished
              else
                   DMASTATE <= DMAWRIT2;

              end if;
         ----------------------------------------


         when DMAEND =>
                   DMASTATE <= DMAEND1; -- Current DMA transfer finished

         when DMAEND1 =>
                   DMASTATE <= DMAEND2; -- Current DMA transfer finished

         when DMAEND2 =>
                   DMASTATE <= DMAEND3; -- Current DMA transfer finished

         when DMAEND3 =>
              --if nHOSTDAK='0' and EPPWAIT='0' and D12DREQ ='1'
              if EPPWAIT='0' and D12DREQ ='1'  -- Sep28, DMA continues
                        and (DCZERO ='0' or DCOUNT(0)='1') then

                   if DMARD_WR='1' then --
```

```
                        DMASTATE <= DMAREAD1; -- Continue the Burst read
mode

                    else --
                        DMASTATE <= DMAWRIT1; -- continue the Burst Wrire
Mode
                    end if;

                elsif EPPWAIT='0' then


                    DMASTATE <= IDLE; -- Idle state

                end if;


            when OTHERS =>
        end case;

    end if;

end process STATE;
---*************************************************************************


---Internal Counter operation*******************************************************
counter: process(CLK,RESET)
begin
    if RESET='1' then
        DCOUNT(BITW downto 0)   <= BITWZERO1; -- Init value


    elsif CLK'event and CLK='1' then
        if DMA_ON ='1'  and DMASTATE=DMAEND then  -- DMA on Cycle
            DCOUNT<=DCOUNT-'1'; --after transfer one byte , counter -1
        --else -- DMA off Cycle
        elsif       DMA_ON='0' and nMCUWR='0' then
            case  LA is  -- Address input
                when "00" => -- Address 00
                    DCOUNT(7 downto 0) <=DIN(7 downto 0); -- Counter Low
Byte

                when "01" => -- Address 01
                    DCOUNT(BITW downto 8) <=DIN((BITW-8) downto 0);--
Counter High Byte

                when others =>

            end case;
        end if;
    end if;
end process counter;
---*************************************************************************


---*************************************************************************
---*************************************************************************
--Data bus  for read/write from/to internal counter
DOUT<=     DCOUNT(7 downto 0) when LA="00" else
           BITWZERO &  DCOUNT(BITW downto 8) ;

--- Decoding Logic ----------------------------------
----States and D12 control signals and EPP signals -----
    nD12DACK<=DMASTATE(3);-- D12 DMA Ack

    -- Internal Dstrobe signal
    nDSTROB<=DMASTATE(2);-- Internal /DSTROBE

    -- External Dstrobe signal
    nDSTROBE<=DMASTATE(2) when DMA_ON='1'  else 'Z';
```

```
    --- D12 read/Write control
    nD12RD      <=DMASTATE(1) when DMA_ON='1'  else nMCURD;--
    nD12WR      <=DMASTATE(0) when DMA_ON='1'  else nMCUWR;--

    ----EPP Write signals -----
    --nEPPWR    <= not DMARD_WR when DMA_ON='1' else 'Z' ;

    ----DMA End of Transfer Signal ----
    nD12EOT     <='0' when nDSTROB='0' and DCZERO='1' and DCOUNT(0)='1'
else '1';

    ---- Counter decreasement or preser(load) Clock-----
    --RWCLK<='0' when (nDSTROB='0' and nD12DACK='0') or
    --          (nMCUWR='0' and nD12DACK='1')  else '1';  --

    --RWCLK<=CLK when (nMCUWR='0' and DMA_ON='0') else
    --          '0' when (nDSTROB='0' and DMA_ON='1')  else '1';  --

    ---Control return to 0/1 control ------
    DCZERO      <='1' when DCOUNT(BITW downto 1) = BITWZERO2 else '0';

    ----Data  Input/output Bus control ------
    DIN       <= DATA;
    --DATA <= DOUT when nMCURD='0' and ALE='0' and LA(1)='0' else
"ZZZZZZZZ";
    DATA    <= DOUT when nMCURD='0' and DMA_on ='0' and LA(1)='0' else
"ZZZZZZZZ";



    ---DMA_On means that DMA Cycle is carrying on even nHOASTDAK is high
    --DMA_ON <='0' when (nHOSTDAK='1' and nD12DACK='1') else '1';
    DMA_ON <='0' when (nHOSTDAK='1' and nD12DACK='1') else '1';


--------------------------------------------------------------------------------
end behave;
```

4.7. Appendix G: PAL (16L8) equations for the main board

As the 32 micro-cells CPLD is not enough to fit the following PAL equations, one additional PAL device (Lattice or AMD PAL18L8) is used to implement this simple coding. The equations are as follows:

```
Name     USBEPp;
Partno   CA0016;
Date     06/07/98;
Rev      01;
Designer  Steven Cheng;
Company   Philips Semiconductors ;
Assembly  None;
Location  None;
Device   g16V8;
/*****************************************************************/
/*                                                               */
/* USB-EPP Eval Board decoding                                   */
/*                                                               */
/*****************************************************************/
/* Allowable Target Device Types :  PAL16L8                      */
/*****************************************************************/
/** Inputs **/
Pin 1 = D12DREQ;    /* D12 DMA request */
/*Pin 2 = DMA_ON;  */
Pin 3 = D0;         /* Data Bus D0 */
Pin 4 = D1;         /* Data Bus D1*/
Pin 5 = SUSPEND;    /* D12 Suspend Mode */
Pin 6 = ASTROBE;    /* EPP ASTROBE */
Pin 7 = HOSTDAK;    /* Host DMA ACK */
Pin 8 = EPPWR;      /* EPP Write */
Pin 9 = DSTROBE;    /* EPP DSTROBE */
Pin 11 = ALE;       /* ADdress Latch */
```

```
/** Outputs **/
pin 12 = D12CS; /* D12 Chip select */
Pin 13 = HOSTDREQ;    /* Host DMA Request      */
/*Pin 14 = DSTROBE;      EPP nDSTROBE, MCU Control */
Pin 15 = LA0;         /* Address A0                */
Pin 16 = LA1;         /* Address A1    */
pin 17 = SHDN;        /* ADM222 Shut down */
Pin 18 = EPPDIR;      /*    /*  EPP 1284 buffer direction control */
Pin 19 = D12A0;       /* D12 Address A0*/

/*pin 19 = */

/* Adder-slice circuit - add 2, 1-bit, numbers with carry */
/* Perform 4, 1-bit, additions and keep the final carry */

LA0 = (D0 & ALE) # (LA0 & !ALE) ;
LA1 = (D1 & ALE) # (LA1 & !ALE) ;
HOSTDREQ = D12DREQ;
D12A0  = LA0 & LA1 & !ALE ;
!D12CS = LA1 & !ALE & HOSTDAK;
!SHDN  = SUSPEND;
EPPDIR  = (EPPWR & !DSTROBE) #
(HOSTDAK & EPPWR & !ASTROBE) ;
```