# DPT5/DPT6
# Desktop PCI
# Telephony

## Hardware Manual

Version 1.5.6

**Communication Automation Corporation**
**1180 McDermott Drive**
**West Chester, PA 19380**

Tel: 610 692-9526
Fax: 610 436-8258
Email: techsupport@cacdsp.com

**Created on 1/14/2009 2:35 PM**

West Chester, PA (USA)

## License Agreement

This Software/Hardware is protected by the copyright laws that pertain to computer software and hardware. It is illegal to duplicate the design and implementation of the Hardware and/or to make copies of the Software except for backups. It is illegal to give copies to another person, or to duplicate the Software by any other means, including electronic transmission. The Software contains trade secrets and, to protect them, you may not decompose, reverse engineer, disassemble, or otherwise reduce the applicable portions of the Software to human perceivable form. You are permitted, however, to freely distribute your own derived software that communicates to the CAC DSP boards through these drivers, free of any royalty.

## Warranty

Communication Automation Corporation reserves the right to make changes to these products, including any software and/or hardware described herein, without notice. No warranty of merchantability or fitness for a particular purpose is expressed or implied. CAC shall not be held liable for incidental or consequential damages in connection with, or arising out of, the use of this Software. CAC does not recommend the use of any of its products, Software or Hardware, for medical or life support applications wherein a failure or malfunction of the product may threaten life or cause injury and will not knowingly license or sell its products for either such use. No rights under any patent accompany the sale of any such products.

## Trademarks

MIPS is a registered trademark of MIPS Technologies, Inc.

POSIX is a registered trademark of the Institute of Electrical and Electronics Engineers, Inc.

VME and VMEbus are registered trademarks of Motorola, Inc.

SCSA is a registered trademark of Dialogic Corporation.

Unix is a registered trademark of Santa Cruz Operations.

GNUPro is a registered trademark of Cygnus Solutions.

smPCI and QuicKit Telephony are trademarks of Communication Automation Corporation.

SPARC, SunOS and Solaris are registered trademarks of Sun Microsystems Computer Corporation and SunSoft.

MS-DOS, Windows95, WindowsNT and Windows are registered trademarks of Microsoft Corporation in the United States of America and other countries.

Use of a term in this manual should not be regarded as affecting the validity of any trademark or service mark.

# Table of Contents

# TABLES

# FIGURES

# CAC/WMI DPT5/6 HARDWARE REFERENCE

## 1. Overview

The DPT5/6 board provides an interface to four E1 or T1 lines in a short form factor standard PCI Bus card. Embedded hardware provides full channelization (with channel bonding), with 128 Mbytes of local SDRAM data buffering. An smPCI™ Module interface allows the E1/T1 serial data to connect to a local TDM bus, which allows data interconnection with two local CAC smPCI™ Module sites. These two sites can house a variety of available modules, adding further processing power. For example, using two CAC DM5420 modules would add the power of 48 TI 5420 DSPs. E1/T1 connectors are EMI shielded RJ45 types. The E1 receiver impedance can be 75 or 120 ohms. The T1 receiver impedance is 100 ohms.

For those familiar with the CAC DPT4, the DPT5/6 is essentially a second generation design of the DPT4. The intent of the DPT5/6 is to provide the same functionality and user interface as that of the DPT4, while enhancing the design is several ways, including PCI/PCI-X compatibility (all speeds), and a more reliable and lower power design. Note that the DPT5/6 does NOT have the H.100 interface that the DPT4 had.

The DPT5 has a 32-bit, 33/66MHz PCI interface, and the DPT6 has an X1 PCIe (Express) interface. Other than the low level differences associated with the PCI/PCIe bridge devices, the two boards function identically and look identical to the host except for a 2-bit field in a status register which may be read to differentiate the two. Henceforth these two boards are referred to as DPT5/6. The photographs below show the DPT5 and DPT6 boards without smPCI modules attached.

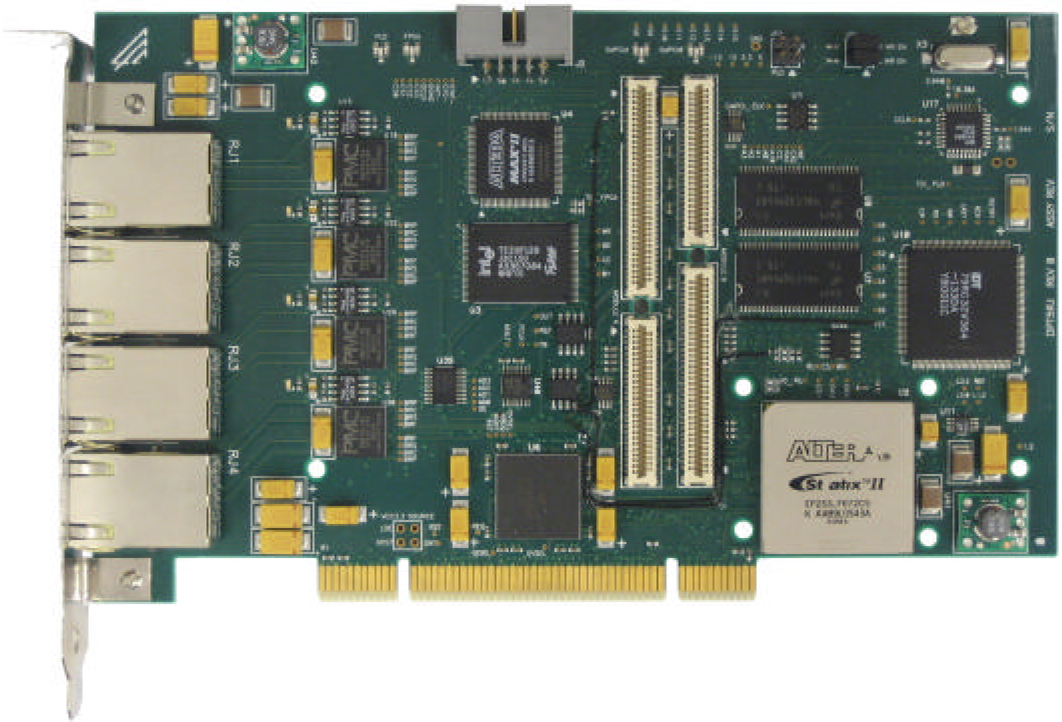


**Figure 1a: DPT5 Board**

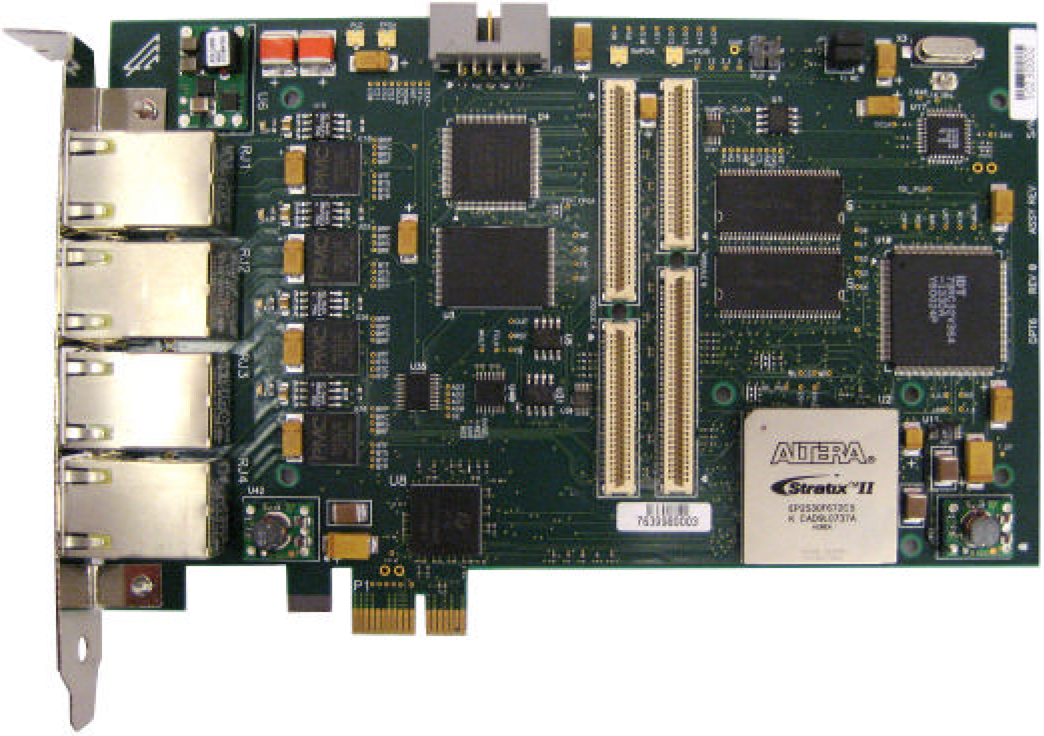


Figure 1b: DPT6 Board

## 2. Block Diagram

A block diagram of the DPT5/6 is shown here:

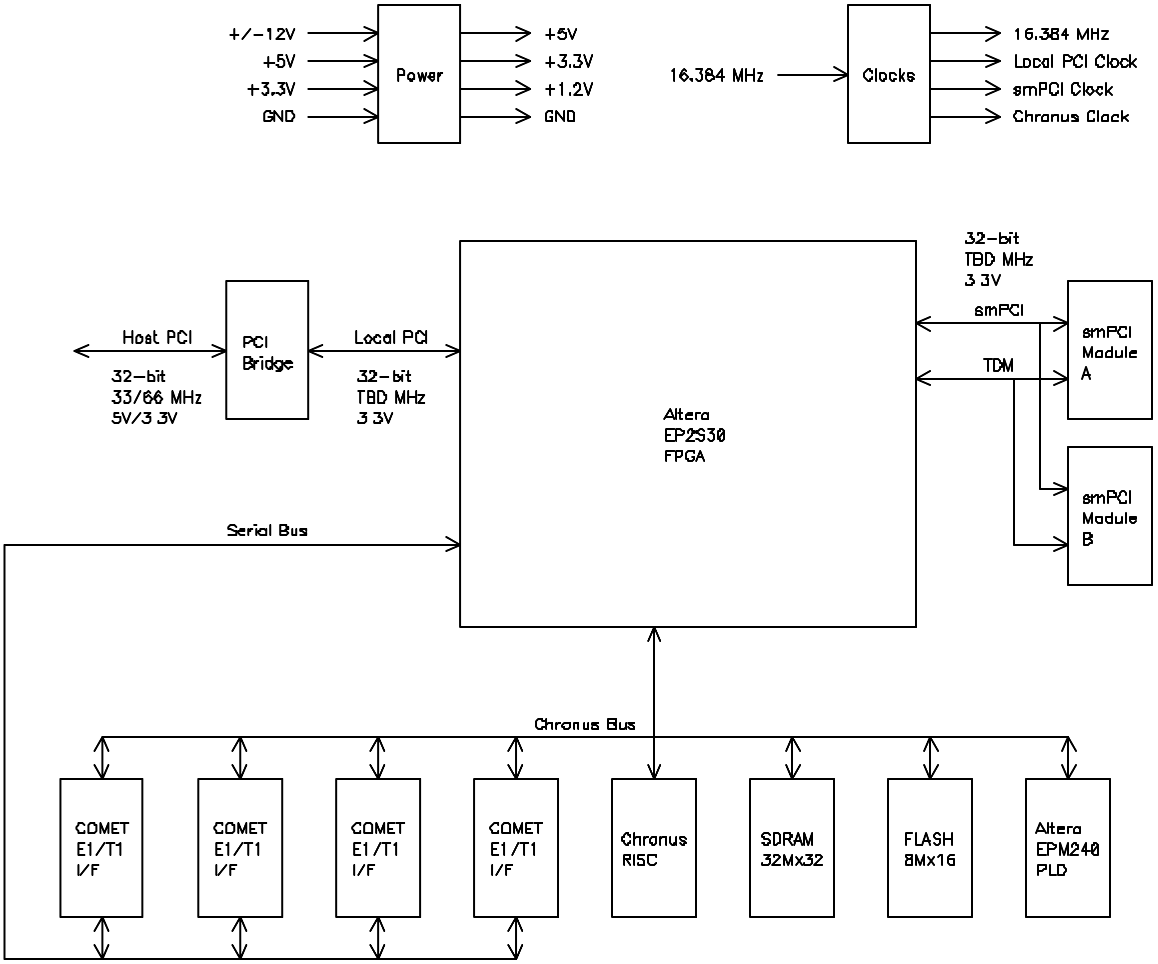


Figure 2: Block Diagram

## 3. Functional Description

Referring to Figure 2, the PCI Bus interface is implemented with a Pericom PI7C8148B PCI bridge. The local side of this bridge connects to an Altera Stratix II FPGA. This FPGA provides further bridging from the local PCI bus to the local Chronus and smPCI buses. On the Chronus bus are the Chronus (a MIPS processor), 32Mx32 SDRAM, 8Mx16 Flash, an Altera PLD, and four PMC-Sierra COMETs (E1/T1 framers). The Altera EPM240 PLD loads the FPGA configuration from Flash at power-on. A 3-PLL clock synthesizer provides system clocks, and switching power supplies create local +3.3V and +1.2V supplies from the PCI Bus +5V input.

# 4. Memory Maps

## 4.1 Host PCI

This section describes how the DPT5/6 resources are mapped from the viewpoint of the host. The DPT5/6 occupies 5 PCI BARs as shown in this table:

**Table 1: Host BAR Mapping**

| BAR | Area | Size (bytes) |
|---|---|---|
| 0 | Peripheral | 256K |
| 1 | SDRAM | 128M |
| 2 | Flash | 16M |
| 3 | SmPCI Module A | 32M(1) |
| 4 | SmPCI Module B | 32M(1) |

(1) For Modules that occupy a space larger than 32M, the host can access the entire space 32M at a time with the use of translation registers in the Host FPGA Register space.

BAR 0 contains several peripherals which are mapped as follows:

**Table 2: BAR 0 Mapping**

| Peripheral | Size (bytes) | Address (offset into BAR0) |
|---|---|---|
| Bank 0 Registers | 1024 | 00000-000FF |
| Bank 1 Registers | 1024 | 01000-010FF |
| Bank 2 Registers | 1024 | 02000-020FF |
| Bank 3 Registers | 1024 | 03000-030FF |
| smPCI Bridge Config Space | 1024 | 03100-031FF |
| Module A Config Space | 1024 | 03200-032FF |
| Module B Config Space | 1024 | 03300-033FF |
| E1/T1 Framer 0 | 256 | 12000-121FF |
| E1/T1 Framer 1 | 256 | 12800-129FF |
| E1/T1 Framer 2 | 256 | 13000-131FF |
| E1/T1 Framer 3 | 256 | 13800-139FF |
| TDM Control RAM | 1024 | 18000-183FF |
| Channelizer DPRAM | 128K | 20000-3FFFF |

## 4.2 Chronus

### 4.2.1 Physical Address

This section describes how the DPT5/6 resources are mapped on the physical Chronus bus.

**Table 3a: Chronus Physical Map**

| Peripheral | Size (bytes) | Physical Address |
|---|---|---|
| SDRAM | 16M | `00000000-0FFFFFFF` |
| Bank 0 Registers | 1024 | `10000000-100000FF` |
| Bank 1 Registers | 1024 | `10001000-100010FF` |
| Bank 2 Registers | 1024 | `10002000-100020FF` |
| Bank 3 Registers | 1024 | `10003000-100030FF` |
| Channelizer DPRAM | 128K | `10020000-1003FFFF` |
| E1/T1 Framer 0 | 256 | `14002000-140021FF` |
| E1/T1 Framer 1 | 256 | `14002800-140029FF` |
| E1/T1 Framer 2 | 256 | `14003000-140031FF` |
| E1/T1 Framer 3 | 256 | `14003800-140039FF` |
| TDM Control RAM | 1024 | `18000000-18001FFF` |
| Flash | 16M | `1FC00000-1FDFFFFF` |
| smPCI Bridge Config Space | 256 | `30003100-300031FF` |
| Module A Config Space | 256 | `30003200-300032FF` |
| Module B Config Space | 256 | `30003300-300033FF` |
| Module A Mem Space | 32M | `40000000-5FFFFFFF` |
| Module B Mem Space | 32M | `60000000-7FFFFFFF` |

### 4.2.2 Logical Address

This section describes how the DPT5/6 resources are mapped on the logical Chronus bus.

**Table 3b: Chronus Logical Map**

| Peripheral | Size (bytes) | Kseg0 (cached, unmapped) | Kseg1 (uncached, unmapped) |
|---|---|---|---|
| SDRAM | 16M | 80000000-8FFFFFFF | A0000000-AFFFFFFF |
| Bank 0 Registers | 1024 | 90000000-90000FFF | B0000000-B0000FFF |
| Bank 1 Registers | 1024 | 90001000-900010FF | B0001000-B00010FF |
| Bank 2 Registers | 1024 | 90002000-900020FF | B0002000-B00020FF |
| Bank 3 Registers | 1024 | 90003000-900030FF | B0002000-B00020FF |
| Channelizer DPRAM | 128K | 90020000-9003FFFF | B0020000-B003FFFF |
| E1/T1 Framer 0 | 256 | 94002000-940021FF | B4002000-B40021FF |
| E1/T1 Framer 1 | 256 | 94002800-940029FF | B4002800-B40029FF |
| E1/T1 Framer 2 | 256 | 94003000-940031FF | B4003000-B40031FF |
| E1/T1 Framer 3 | 256 | 94003800-940039FF | B4003800-B40039FF |
| TDM Control RAM | 1024 | 98000000-98001FFF | B8000000-B8001FFF |
| Flash | 16M | 9FC00000-9FDFFFFF | BFC00000-BFDFFFFF |
| smPCI Bridge Config Space | 256 | TBD | TBD |
| Module A Config Space | 256 | TBD | TBD |
| Module B Config Space | 256 | TBD | TBD |
| Module A Mem Space | 32M | TBD | TBD |
| Module B Mem Space | 32M | TBD | TBD |

## 4.3 smPCI Module

This section describes how the DPT5/6 resources are mapped from the viewpoint of the CAC smPCI Modules.

**Table 3c: Module Map**

| Peripheral | Size (bytes) | Address |
|---|---|---|
| SDRAM | 16M | `40000000-4FFFFFFF` |
| Bank 0 Registers | 1024 | TBD |
| Bank 1 Registers | 1024 | TBD |
| Bank 2 Registers | 1024 | TBD |
| Bank 3 Registers | 1024 | TBD |
| smPCI Bridge Config Space | 256 | TBD |
| Module A Config Space | 256 | TBD |
| Module B Config Space | 256 | TBD |
| Channelizer DPRAM | 128K | TBD |
| E1/T1 Framer 0 | 256 | TBD |
| E1/T1 Framer 1 | 256 | TBD |
| E1/T1 Framer 2 | 256 | TBD |
| E1/T1 Framer 3 | 256 | TBD |
| TDM Control RAM | 1024 | TBD |
| Flash | 16M | TBD |

## 4.4 Host to smPCI Mapping

This section describes how Host PCI bus addresses are translated to Chronus bus and smPCI bus addresses for selected smPCI access types. There are three smPCI configuration spaces: the smPCI Bridge, Module A, and Module B. All three spaces are accessed via BAR 0, using offsets of 0x3100, 0x3200 and 0x3300, respectively. The Host Bridge FPGA translates the Host PCI address offset into a Chronus physical address as follows:

**Table 4a: Host to smPCI Config Space Address Translation**

| Chronus Address | Value |
|---|---|
| a[31:28] | 0011 |
| a[27:23] | 01000* for smPCI bridge config space<br>00010* for smPCI Mod A config space<br>00001* for smPCI Mod B config space |
| a[22:8] | All 0's |
| a[7:2] | Copied from host PCI AD[7:2] |
| a[1:0] | All 0's |

*Note that a26, a24 and a23 are used is IDSELs on the smPCI bus for these spaces. The smPCI address is formed by replacing the upper 4 bits with 0's and passing the rest of the address bits through.

The memory spaces for the two smPCI modules are accessed via BARs 3 and 4, respectively. BARs 3 and 4 are given a 32 Mbyte space on the Host PCI bus. Therefore, to access module memory space beyond 32 Mbytes, a 4-bit translation register for each module is provided to select one of sixteen 32-Mbyte sections of the module memory space. The table below shows how the Chronus address is formed.

**Table 4b: Host to smPCI Memory Space Address Translation**

| Chronus Address | Value |
|---|---|
| A[31:29] | 010 for Mod A<br>011 for Mod B |
| A[28:25] | Copied from the translation register |
| A[24:2] | Copied from host PCI AD[24:2] |
| A[1:0] | All 0's |

The smPCI address is formed by replacing the upper 4 bits with 6 (for Mod A) or 8 (for Mod B) and passing the rest of the address bits through.

The table below shows some examples of the address translations from Host to smPCI buses. Key bits are bolded.

**Table 4c: Host to smPCI Addressing**

| Host BAR | Access Type/ Host Offset | Chronus Address | SmPCI Address |
|---|---|---|---|
| 0 | SmPCI Bridge Cfg Space<br>xxxx xxxx xxxx xxxx<br>**0011 0001** aaaa aaxx | **0011** 0**1**00 0000 0000<br>0000 0000 hhhh hh00 | **0000** cccc cccc cccc<br>cccc cccc cccc cc00 |
| 0 | SmPCI Mod A Cfg Space<br>xxxx xxxx xxxx xxxx<br>**0011 0010** aaaa aaxx | **0011** 000**1** 0000 0000<br>0000 0000 hhhh hh00 | **0000** cccc cccc cccc<br>cccc cccc cccc cc00 |
| 0 | SmPCI Mod B Cfg Space<br>xxxx xxxx xxxx xxxx<br>**0011 0011** aaaa aaxx | **0011** 0000 **1**000 0000<br>0000 0000 hhhh hh00 | **0000** cccc cccc cccc<br>cccc cccc cccc cc00 |
| 3 | SmPCI Mod A Mem Space<br>xxxx xxx**a aaaa aaaa**<br>**aaaa aaaa aaaa aa**xx | **010r rrr**h hhhh hhhh<br>hhhh hhhh hhhh hh00 | **0110** cccc cccc cccc<br>cccc cccc cccc cc00 |
| 4 | SmPCI Mod B Mem Space<br>xxxx xxx**a aaaa aaaa**<br>**aaaa aaaa aaaa aa**xx | **011r rrr**h hhhh hhhh<br>hhhh hhhh hhhh hh00 | **1000** cccc cccc cccc<br>cccc cccc cccc cc00 |

Notes

1. a = address bit, x = don’t care bit
2. h = copied from Host address, c = copied from Chronus address
3. r = copied from the translation register for that module. The field select one of 16 32Mbyte sections of the module memory (for those modules with a memory space greater than 32 Mbytes). See Section 5.3.4.

The table below summarizes the address translations (for rrrr = 0).

**Table 4d: Host to smPCI Address Translation Summary**

| Host BAR | Host Offset | Chronus Address | SmPCI Address |
|---|---|---|---|
| 0 | 0x31xx | 0x340000xx | 0x040000xx |
| 0 | 0x32xx | 0x310000xx | 0x010000xx |
| 0 | 0x33xx | 0x308000xx | 0x008000xx |
| 3 | 0x01234567 | 0x41234567 | 0x61234567 |
| 4 | 0x01234567 | 0x61234567 | 0x81234567 |

# 5. Control Registers

## 5.1 Bank 0 Registers (formerly “Host FPGA Registers”)

The Bank 0 Registers start in Bar 0 at offset 0x0000 and are as follows. Shaded cells indicate new registers for DPT5/6.

**Table 5: Bank 0 Registers**

| Offset | 0 | 4 | 8 | C |
|---|---|---|---|---|
| 0x00 | | | | |
| 0x10 | | | | |
| 0x20 | | | | |
| 0x30 | Set IRQ | Clear IRQ | Set IRQ Mask | Clear IRQ Mask |
| 0x40 | JTAG Ctrl | Arbiter Control | Clock Synth Ctrl | Hardware Ctrl |
| 0x50 | L1 Rx Ctrl | Rx Frame Cnt | L1 Tx Ctrl | Tx Frame Cnt |
| 0x60 | DMA Ctrl | DMA Address | DMA Test Data | DMA Cnt |
| 0x70 | Hardware Ctrl 2 | L1 Timing Ctrl | COMET I/F Ctrl | |
| 0x80 | | | | |
| 0x90 | Freq Cntr Ctrl | Freq 0 | Freq 1 | Freq 2 |
| 0xA0 | I2C Ctrl | I2C Data | I2C Status | |
| 0xB0 | | | | |
| 0xC0 | Misc Status | | | |
| 0xD0 | | | | |
| 0xE0 | | | | |
| 0xF0 | | | DPT5/6 ID | Rev Code |

### 5.1.1 Set IRQ (0x30), Clear IRQ (0x34)

This pair of 32-bit registers is used to control and monitor interrupts to the host and to the Chronus. Writing a ‘1’ to any writable bits in the SET IRQ register will set those bits. Writing a ‘1’ to any writable bits in the CLR IRQ register will clear those bits. Zeros written to either register have no effect. This behavior eliminates problems arising from collisions of read-modify-write operations initiated by multiple processes. Reading either register will give the same interrupt status. Bits are assigned as follows:

**Table 8: Set IRQ/Clear IRQ Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:16 | Reserved | 0 | |
| 15:12 | Reserved | Tx IRQ | 1 = asserted |
| 11:8 | Reserved | Rx IRQ | 1 = asserted |
| 7 | Reserved | Chronus INT2 | 1 = asserted |
| 6 | Reserved | Chronus INT3 | 1 = asserted |
| 5 | Reserved | NMI Status | 1 = asserted |
| 4 | Reserved | Reset IRQ | 1 = asserted |
| 3:2 | IRQ to Host | IRQ to Host | 1 = asserted |
| 1:0 | IRQ to Chronus | IRQ to Chronus | 1 = asserted |

### 5.1.2 Set IRQ Mask (0x38), Clear IRQ Mask (0x3C)

This pair of 32-bit registers is used to control interrupt masks. Writing a ‘1’ to any writable bits in the SET IRQ MASK register will set those bits. Writing a ‘1’ to any writable bits in the CLR IRQ MASK register will clear those bits. Zeros written to either register have no effect. This behavior eliminates problems arising from collisions of read-modify-write operations initiated by multiple processes. Reading either register will give the same interrupt mask. Bits are assigned as follows:

**Table 9: Set IRQ Mask/Clear IRQ Mask Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:17 | Reserved | 0 | |
| 16 | PCI Int Disable | PCI Int Disable | 1 = disabled |
| 15:13 | Reserved | 0 | |
| 12 | Tx IRQ Mask | Tx IRQ Mask | 1 = enabled |
| 11:8 | Rx IRQ Mask | Rx IRQ Mask | 1 = enabled |
| 7 | Enable Chronus INT2 to Host | Enable Chronus INT2 to Host | 1 = enabled |
| 6 | Enable Chronus INT3 to Host | Enable Chronus INT3 to Host | 1 = enabled |
| 5 | Reserved | 0 | |
| 4 | Reset IRQ | Reset IRQ | 1 = enabled |
| 3:2 | IRQ to Host | IRQ to Host | 1 = enabled |
| 1:0 | IRQ to Chronus | IRQ to Chronus | 1 = enabled |

### 5.1.3 JTAG Control (0x40)

This 32-bit register provides control for the DPT5/6 JTAG chain. See Section 11.

**Table 10: JTAG Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:11 | Reserved | 0 | |
| 10 | Reserved | JTAG Cable | 1 = JTAG cable is connected |
| 9 | JTAG OE | JTAG OE | 1 = enabled |
| 8 | Reserved | TDO | JTAG TDO |
| 7 | Busy | Busy | 1 = JTAG is busy |
| 6 | Auto TCk Mode | Auto TCk Mode | 1 = Auto TCK Mode |
| 5 | EECS | EECS | Selects DPT5/6 EEPROM |
| 4 | TMS | TMS | Selects DPT5/6 JTAG Chain |
| 3 | TMSA | TMSA | Selects SmPCI Module A JTAG Chain |
| 2 | TMSB | TMSB | Selects SmPCI Module B JTAG Chain |
| 1 | TCK | TCK | JTAG TCK |
| 0 | TDI | TDI | JTAG TDI |

This register also allows control of the DPT5/6 serial EEPROM, which is not a JTAG device, but shares some of the control signals.

### 5.1.4 Arbiter Control (0x44)

This 32-bit register controls the MIPS bus arbiter.

**Table 11a: Arbiter Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:9 | Reserved | 0 | |
| 8 | Delay Enable | Delay Enable | 0=no delay, 1=delay |
| 7-0 | Delay Time | Delay Time | Arbiter delay time = (N+1)*15ns |

### 5.1.5 Clock Synthesizer Control (0x48)

This 32-bit register no longer configures the clock synthesizer, which is now on the JTAG bus. It still contains two status bits.

**Table 11b: Clock Synthesizer Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:8 | Reserved | 0 | |
| 7 | Flashup Flag | Flashup Flag | 1 = flashed |
| 6 | Init Flag | Init Flag | 1 = Init'd |
| 5-0 | Reserved | 0 | |

### 5.1.6 Hardware Control (0x4C)

This 32-bit register is used to control a variety of functions.

**Table 12: Hardware Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31 | Rsvd | 1 | 1 = Has reg lock |
| 30 | Rsvd | 0 | Always 0 |
| 29:26 | Rsvd | 0 | |
| 25:24 | Reg Access Lock | Reg Access Lock | (1) |
| 23:20 | Module A Translate | Module A Translate | (2) |
| 19:16 | Module B Translate | Module B Translate | (2) |
| 15 | Flash Write Enable | Flash Write Enable | 1 = enabled |
| 14 | Disable Refresh | Disable Refresh | 1 = disabled |
| 13 | Deassert Reset | Deassert Reset | 0 = asserted |
| 12 | Deassert Cold Reset | Deassert Cold Reset | 0 = asserted |
| 11 | Deassert Local PCI reset | Deassert Local PCI reset | 0 = asserted |
| 10 | MUXBT Hi-Z | MUXBT Hi-Z | 1 = Hi-Z |
| 9 | LED | LED | 0 = On |
| 8 | Flash Sync Mode | Flash Sync Mode | 1 = sync |
| 7 | BERR enable | BERR enable | 1 = enabled |
| 6 | MUXBTCLK Hi-Z | MUXBTCLK Hi-Z | 1 = Hi-Z |
| 5 | NMI enable | NMI enable | 1 = enabled |
| 4 | Rev 0 Mode | Rev 0 Mode | 0 = Rev0, 1 = Rev2 |
| 3 | Rsvd | 0 | |
| 2 | Rsvd | DMA FIFO not full | 0 = DMA FIFO full |
| 1 | Rsvd | DMA busy | 1 = DMA busy |
| 0 | Rsvd | DMA armed | 1 = DMA armed |

Notes:

(1) A process wishing to access this register checks that this field is 0. If so, it writes a non-zero value, thus keeping other processes from accessing it. When the process is finished it writes a zero back to it, freeing it for the next process.

(2) For the case where modules have a space larger than the 32M BAR3/BAR4 space, these four bit fields select one of up to 16 32M pieces of that space.

## 5.1.7 E1/T1 Rx Control (0x50)

This 32-bit register is used to control the E1/T1 receive functions.

**Table 13: E1/T1 Rx Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:28 | Clear Rx Bank IRQ | Rx Bank IRQ | Self-clearing (1) |
| 27:24 | Reserved | Rx Bank | Active Bank (1) |
| 23:20 | Reserved | Rx Overflow | 1 = overflow (1) |
| 19:16 | Reserved | 0 | |
| 15:8 | Frames per buffer-1 | Frames per buffer-1 | |
| 7 | Test Pattern | Test Pattern | Do not use |
| 6 | Unframed Operaion | Unframed Operaion | 1 = unframed |
| 5 | Reserved | 0 | |
| 4 | Rx BP rate | Rx BP rate | 0=2.048, 1=1.544 |
| 3:0 | Rx Enable | Rx Enable | 1 = enabled (1) |

Notes:

(1) These are four bit fields where the LSB corresponds to Framer A and the LSB corresponds to Framer D

### 5.1.8 E1/T1 Rx Frame Counter (0x54)

This 32-bit register is used to read the E1/T1 frame counters.

**Table 13: E1/T1 Rx Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:24 | COMET 0 Frame Cnt | COMET 0 Frame Cnt | 1 |
| 23-16 | COMET 1 Frame Cnt | COMET 1 Frame Cnt | 1 |
| 15-8 | COMET 2 Frame Cnt | COMET 2 Frame Cnt | 1 |
| 7-0 | COMET 3 Frame Cnt | COMET 3 Frame Cnt | 1 |

Notes:

(1) tbd

### 5.1.9 E1/T1 Tx Control (0x58)

This 32-bit register is used to control the E1/T1 transmit functions.

**Table 14: E1/T1 Tx Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:28 | Clear Tx Bank IRQ | Tx Bank IRQ | Self-clearing (1) |
| 27:24 | Reserved | Tx Bank | Active Bank (1) |
| 23:20 | Reserved | Tx Underflow | 1 = overflow (1) |
| 19:16 | Reserved | 0xE | |
| 15:8 | Reserved | 0 | |
| 7 | Reserved | 0 | |
| 6 | Tx Clock Select | Tx Clock Select | 0=Local-based,<br>1=TDM-based |
| 5 | Reserved | 0 | |
| 4 | Reserved | 0 | |
| 3:0 | Tx Enable | Tx Enable | 1 = enabled (1) |

Notes:

(1) These are four bit fields where the LSB corresponds to Framer A and the LSB corresponds to Framer D

### 5.1.10 E1/T1 Tx Frame Counter (0x5C)

This 32-bit register is used to read the E1/T1 frame counters.

**Table 13: E1/T1 Rx Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:24 | COMET 0 Frame Cnt | COMET 0 Frame Cnt | 1 |
| 23-16 | COMET 1 Frame Cnt | COMET 1 Frame Cnt | 1 |
| 15-8 | COMET 2 Frame Cnt | COMET 2 Frame Cnt | 1 |
| 7-0 | COMET 3 Frame Cnt | COMET 3 Frame Cnt | 1 |

Notes:

(1) tbd

### 5.1.11 DMA Control (0x60)

This 32-bit register is used to DMA transfers from SDRAM to the host.

**Table 15: DMA Word Count Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:7 | Reserved | 0 | |
| 5:0 | DMA Word Count | DMA Word Count | (use 0 for 64 words) |

### 5.1.12 DMA Host Address (0x64)

This 32-bit register is used to set the host destination address for DMA transfers from SDRAM to the host. Writing to this register arms the DMA controller, at which point the Chronus may start the DMA by reading a special address.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | DMA Host Address | DMA Host Address | |

### 5.1.13 DMA Test Data (0x68)

This 32-bit register is used to set the starting data for DMA transfers from SDRAM to the host. Data is incremented for each word transferred.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | DMA Test Data | DMA Test Data | |

### 5.1.14 DMA Count (0x6C)

This 32-bit register is used to set or read a count of the number of DMA transfers (in blocks) which have occurred from SDRAM to the host.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | DMA Count | DMA Count | |

### 5.1.15 Hardware Ctrl 2 (0x70)

This 32-bit register is used for various control.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:16 | Reserved | 0 | |
| 15-12 | DMA Ctrl | DMA Ctrl | |
| 11 | TDO Mux Ctrl | TDO Mux Ctrl | |
| 10 | Split JTAG | Split JTAG | |
| 9 | smPCI Test | smPCI Test | |
| 8 | DMA Reset | DMA Reset | |
| 7 | Reserved | 0 | |
| 6 | Kill smPCI Refresh | Kill smPCI Refresh | |
| 5 | Chronus PCST Ctrl | Chronus PCST Ctrl | |
| 4 | Chronus Int Ctrl | Chronus Int Ctrl | |
| 3-2 | PLD Ctrl | PLD Ctrl | Self-clearing |
| 1 | FastClock Mult | FastClock Mult | |
| 0 | SDRAM Ctrlr Reset | SDRAM Ctrlr Reset | Self-clearing |

Notes:

| | |
|---|---|
| Bits 15-12: | Used for DMA test |
| Bit 11: | Set to force TDO mux to select TDO (not TDO2) |
| Bit 10: | Set to indicate that the board has a split JTAG bus |
| Bit 9: | Reserved |
| Bit 8: | Set to reset DMA logic |
| Bit 6: | Set to suppress smPCI refresh |
| Bit 5: | Reserved |
| Bit 4: | Reserved |
| Bit 3-2: | PLD Control, both are self-clearing. Set bit 3 to tell the PLD to reconfigure the FPGA from flash bank 1. Set bit 2 to tell the PLD to reconfigure the FPGA from flash bank 0. |
| Bit 1: | 0 = 16x, 1 = 24x |
| Bit 0: | Set to reset SDRAM controller logic. Self-clearing. |

### 5.1.16 L1 Timing Control (0x74)

This 32-bit register is used for various control.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | Reserved | 0 | |

Notes:

tbd

### 5.1.17 COMET Interface Control (0x78)

This 32-bit register is used for various control.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:8 | Reserved | 0 | |
| 7-6 | BT* Mux Ctrl | BT* Mux Ctrl | |
| 5-4 | BR* Mux Ctrl | BR* Mux Ctrl | |
| 3-1 | Reserved | 0 | |
| 0 | E1/T1 Mode | E1/T1 Mode | |

Notes:

Bit 7-6: BT* Mux Ctrl:

- 00: BT* driven by channelizer
- 01: BT* driven by BR*
- 10: BT* driven by BXGEN8
- 11: BT* driven by BXGEN32

Bit 5-4: BR* Mux Ctrl:

- 00: internal BR* driven by COMET BR*
- 01: internal BR* driven by COMET BT*
- 10: internal BR* driven by BXGEN32*
- 11: internal BR* driven by BXGEN8*

Bit 0: 0=E1, 1=T1

### 5.1.18 Frequency Counter Control (0x90)

This 32-bit register is used for frequency counter control.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:19 | Reserved | 0 | |
| 18:16 | Reserved | FC Ready | 1 |
| 15:2 | Reserved | 0 | |
| 1:0 | Reserved | FC Scale | |

Notes:

Bit 18: A 1 indicates that the Freq 2 value is valid

Bit 17: A 1 indicates that the Freq 1 value is valid

Bit 16: A 1 indicates that the Freq 0 value is valid

Bit 1-0: Sets the scale factor for the Freq 2-0 values as follows:

00: Actual frequency = Freq N * 1

01: Actual frequency = Freq N * 10

10: Actual frequency = Freq N * 100

11: Actual frequency = Freq N * 1000

### 5.1.19 Frequency 0 (0x94)

This 32-bit register is used to read the local PCI bus clock frequency. See Reg 0x90 for scaling information.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | Reserved | PCI Clock Frequency | |

Notes:

tbd

### 5.1.20 Frequency 1 (0x98)

This 32-bit register is used to read the Chronus bus clock frequency. See Reg 0x90 for scaling information.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | Reserved | Chronus Clock Frequency | |

Notes:

Tbd

### 5.1.21 Frequency 2 (0x9C)

This 32-bit register is used to read the local smPCI clock frequency. See Reg 0x90 for scaling information.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | Reserved | smPCI Clock Frequency | |

Notes:

tbd

### 5.1.22 Misc Status (0xC0)

This 32-bit register is used to read the various status.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:15 | Reserved | Spare Pin States | |
| 15:14 | Reserved | 0 | |
| 13:12 | Reserved | PCB Type | 1 |
| 11:10 | Reserved | 0 | |
| 9 | Reserved | Bridge Enum | 2 |
| 8 | Reserved | Bridge LOO | 3 |
| 7:4 | Reserved | Bridge GPIO | 4 |
| 3 | Reserved | PLL Loss of Clock | 5 |
| 2 | Reserved | Low Voltage – 5V | 6 |
| 1 | Reserved | Low Voltage – 3.3V | 6 |
| 0 | Reserved | Low Voltage – 1.2V | 6 |

Notes:

1) PCB Types: 00=DPT6, 10=DPT6RJ, 10=DPT5, 11=DPT5RJ
2) Indicates PCI bridge ENUM pin state.
3) Indicates PCI bridge LOO pin state.
4) Indicates PCI bridge GPIO pin states.
5) Indicates PLL LOS pin state.
6) Indicates an under-voltage condition on the indicated power supply.

### 5.1.23 Device (0xF8)

This 32-bit register is used to read the DPT5/6 device ID.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | Reserved | 0x000005E1 | |

Notes:

Tbd

### 5.1.24 FPGA Revision Code (0xFC)

This 32-bit register is used to read the FPGA revision code, which is set at FPGA build time.

**Table 16: DMA Host Address Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | Reserved | FPGA Rev Code | |

Notes:

tbd

## 5.2 Bank 1 Registers (formerly “Expansion FPGA Registers”)

The Bank 1 Registers start in Bar 0 at offset 0x1000 and are as follows:

**Table 5.2: Bank 1 Registers**

| Offset | 0 | 4 | 8 | C |
|---|---|---|---|---|
| 0x00 | | | | |
| 0x10 | | | | |
| 0x20 | | | | |
| 0x30 | Set IRQ | Clear IRQ | | |
| 0x40 | | TDM Ctrl | L1 Ctrl | Hardware Ctrl |
| 0x50 | Indir A Addr | Indir A Data | Indir A Ctrl | |
| 0x60 | Indir B Addr | Indir B Data | Indir B Ctrl | |
| 0x70 | | | | |
| 0x80 | | | | |
| 0x90 | | | | |
| 0xA0 | | | | |
| 0xB0 | | | | |
| 0xC0 | | | | |
| 0xD0 | | | | |
| 0xE0 | | | | |
| 0xF0 | | | | |

### 5.2.1 Set IRQ Control (0x1030), Clear IRQ Control (0x1034)

This pair of 32-bit registers is used to control and monitor interrupts to and from the smPCI Modules. Writing a ‘1’ to any writable bits in the SET IRQ Control register will set those bits. Writing a ‘1’ to any writable bits in the CLR IRQ Control register will clear those bits. Zeros written to either register have no effect. This behavior eliminates problems arising from collisions of read-modify-write operations initiated by multiple processes. Reading either register will give the same interrupt status. Bits are assigned as follows:

**Table 18: Set IRQ/Clear IRQ Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:22 | Rsvd | 0 | |
| 21 | Rsvd | Mod A Global Int | 1=asserted |
| 20 | Rsvd | Mod B Global Int | 1=asserted |
| 19 | Rsvd | ModA Int A | 1=asserted |
| 18 | Rsvd | ModA Int B | 1=asserted |
| 17 | Rsvd | ModB Int A | 1=asserted |
| 16 | Rsvd | ModB Int B | 1=asserted |
| 15 | Rsvd | Master error | 1-error (1) |
| 14:10 | Rsvd | 0 | |
| 9 | Mod A Global Int | Mod A Global Int | 1=assert |
| 8 | Mod B Global Int | Mod B Global Int | 1=assert |
| 7 | Mod A Global Int Sel | Mod A Global Int Sel | 0=host, 1=ModB Global |
| 6 | Mod B Global Int Sel | Mod B Global Int Sel | 0=host, 1=ModA Global |
| 5 | Mod A Global Int OE | Mod A Global Int OE | 1=enabled |
| 4 | Mod B Global Int OE | Mod B Global Int OE | 1=enabled |
| 3 | ModA Int A enable | ModA Int A enable | 1=enabled |
| 2 | ModA Int B enable | ModA Int B enable | 1=enabled |
| 1 | ModB Int A enable | ModB Int A enable | 1=enabled |
| 0 | ModB Int B enable | ModB Int B enable | 1=enabled |
| 3:2 | IRQ to Host | IRQ to Host | 1 = asserted |
| 1:0 | IRQ to Chronus | IRQ to Chronus | 1 = asserted |

Notes:

(1) This bit is set when a local PCI master error has occurred on the last master transaction. This includes missing DEVSEL or TRDY timeout errors. A successful master transaction will clear this bit.

### 5.2.2 TDM Control (0x1044)

This 32-bit register is used to control a variety of TDM functions.

**Table 19: TDM Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:24 | Reserved | 0 | |
| 23:16 | Reserved | 0 | |
| 15:13 | Reserved | 0 | |
| 12-8 | Frames per MF - 1 | Frames per MF – 1 | Sets MF rate (2) |
| 7-6 | Reserved | 0 | |
| 5 | Rsvd | bank irq | 1=active |
| 4 | Rsvd | bank | |
| 3 | Rsvd | bank sw pending | 1=active |
| 2 | clr irq | 0 | Self-clearing bit |
| 1 | bank sw | 0 | Self-clearing bit |
| 0 | tdm clr | tdm clr | 1=clear, 0=run |

Notes:

(1) For E1 set this to 15 (16 frames per multiframe). For T1 set this to 11 (D4 mode - 12 frames per multiframe) or 23 (ESF mode - 24 frames per multiframe).

### 5.2.3 E1/T1 Control (0x1048)

This 32-bit register is used to control a variety of E1/T1 functions.

**Table 20: E1/T1 Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:19 | Rsvd | 0 | |
| 18 | FC IRQ Clear | FC IRQ | Self-clearing bit(s) |
| 17:16 | Rsvd | Current FC Select | 0=A, 1=B, 2=C, 3=D |
| 15 | BT Src Select | BT Src Select | 1 |
| 14:13 | Rsvd | 0 | |
| 12 | Slave Mode | Slave Mode | (3) |
| 11:8 | Rsvd | 0 | |
| 7:5 | Timing master select | Timing master select | 0=local, 1=fastclk, 2-7 invalid |
| 4 | E1/T1 mode | E1/T1 mode | 0=E1, 1=T1 |
| 3:0 | Fastclk enables | Fastclk enables | (1) |

Notes:

(1) Set this to drive the COMET BT (transmit) signals from the TDM subsystem.

(2) This is a four bit field where the LSB corresponds to Framer A and the LSB corresponds to Framer D

(3) Set this bit to enable this board to be slaved to another DPT5/6 board. This board’s TDM and E1/T1 transmit clocks will be locked to the other DPT5/6. This mode requires a special board-to-board interconnect provided by CAC. Please call for further support.

### 5.2.4 Hardware Control (0x104C)

This 32-bit register is used to control a variety of functions.

**Table 21: Hardware Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:16 | Rsvd | 0 | |
| 15 | Rsvd | 0 | |
| 14 | Rsvd | 0 | |
| 13 | Rsvd | 0 | |
| 12 | Rsvd | 0 | |
| 11 | Chronus Int3 test | Chronus Int3 test | 1=enabled |
| 10 | Rsvd | 0 | |
| 9 | Rsvd | 0 | |
| 8 | Rsvd | 0 | |
| 7 | Rsvd | 0 | |
| 6 | Rsvd | 0 | |
| 5 | Rsvd | 0 | |
| 4 | PCI module PROG- | PCI module PROG- | 1 = assert |
| 3:2 | Rsvd | 0 | |
| 1 | Rsvd | 0 | |
| 0 | Rsvd | 0 | |

Notes:

### 5.2.5 smPCI Indirect Read A Address (0x1050)

This 32-bit register is used to set the address for indirect smPCI reads.

**Table 21: Hardware Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | Address | Address | |

Notes:

### 5.2.6 smPCI Indirect Read A Data (0x1054)

This 32-bit register is used to set the address for indirect smPCI reads.

**Table 21: Hardware Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | Reserved | Data | |

Notes:

### 5.2.7 smPCI Indirect Read A Control (0x1058)

This 32-bit register is used to set the address for indirect smPCI reads.

**Table 21: Hardware Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:14 | Reserved | 0 | |
| 13 | Reserved | Read Busy | |
| 12 | Reserved | Data Ready | |
| 11:5 | Reserved | 0 | |
| 4 | Auto Addr Incr Mode | Auto Addr Incr Mode | |
| 3:0 | PCI Cmd Code | PCI Cmd Code | |

Notes:

### 5.2.8 smPCI Indirect Read B Address (0x1060)

This 32-bit register is used to set the address for indirect smPCI reads.

**Table 21: Hardware Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | Address | Address | |

Notes:

### 5.2.9 smPCI Indirect Read B Data (0x1064)

This 32-bit register is used to set the address for indirect smPCI reads.

**Table 21: Hardware Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:0 | Reserved | Data | |

Notes:

### 5.2.10 smPCI Indirect Read B Control (0x1068)

This 32-bit register is used to set the address for indirect smPCI reads.

**Table 21: Hardware Control Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:14 | Reserved | 0 | |
| 13 | Reserved | Read Busy | |
| 12 | Reserved | Data Ready | |
| 11:5 | Reserved | 0 | |
| 4 | Auto Addr Incr Mode | Auto Addr Incr Mode | |
| 3:0 | PCI Cmd Code | PCI Cmd Code | |

Notes:

## 5.3 Bank 2 Registers (formerly "PLD Registers")

The Bank 2 Registers start in Bar 0 at offset 0x2000 and are as follows:

**Table 5.3: Bank 2 Registers**

| Offset | 0 | 4 | 8 | C |
|---|---|---|---|---|
| 0x00 | Ctrl 0 | Ctrl 1 | Status | |
| 0x10 | | | | |
| 0x20 | | | | |
| 0x30 | | | | |
| 0x40 | | | | |
| 0x50 | | | | |
| 0x60 | | | | |
| 0x70 | | | | |
| 0x80 | | | | |
| 0x90 | | | | |
| 0xA0 | | | | |
| 0xB0 | | | | |
| 0xC0 | | | | |
| 0xD0 | | | | |
| 0xE0 | | | | |
| 0xF0 | | | | |

### 5.3.1 Control Register 0 (0x2000)

This 32-bit register is used to control a variety of functions.

**Table 22: Control Register 0**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:21 | Rsvd | 0 | |
| 20 | Rsvd | 0 | |
| 19 | Rsvd | 0 | |
| 18 | BR-to-BT loopback | BR-to-BT loopback | 1 = loopback in FPGA |
| 17 | Rsvd | 0 | 0=Txclk,<br>1 = BRCLK |
| 16 | Btclk sel | btclk sel | |
| 15 | Rsvd | 0 | |
| 14 | Rsvd | 0 | |
| 13 | Brclk sel1 | brclk sel1 | 0 = A, 1 = B, 2 = C, 3 = D (1) |
| 12 | Brclk sel0 | brclk sel0 | |
| 11 | Rsvd | 0 | |
| 10 | Rsvd | 0 | |
| 9:0 | Rsvd | 0 | |

Notes:

(1) When the btclk select field selects brclk, this field select which brclk to use for transmit backplane timing.

### 5.3.2 Control Register 1 (0x2004)

This 32-bit register is used to control certain interrupts.

**Table 23: Control Register 1**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:21 | Rsvd | 0 | |
| 20 | Rsvd | 0 | |
| 19 | Rsvd | 0 | |
| 18 | Rsvd | 0 | |
| 17 | Rsvd | 0 | |
| 16 | Rsvd | 0 | |
| 15 | Rsvd | 0 | |
| 14 | Rsvd | 0 | |
| 13 | Comet D Int enable | Comet D Int enable | 1 = enabled |
| 12 | Comet C Int enable | Comet C Int enable | 1 = enabled |
| 11 | Comet B Int enable | Comet B Int enable | 1 = enabled |
| 10 | Comet A Int enable | Comet A Int enable | 1 = enabled |
| 9:0 | Rsvd | 0 | |

### 5.3.3 Status Register (0x2008)

This 32-bit register is used to read interrupt and FPGA configuration status.

**Table 24: Status Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:21 | Rsvd | 0 | |
| 20 | Rsvd | 0 | |
| 19 | Rsvd | 0 | |
| 18 | Rsvd | 0 | |
| 17 | Rsvd | 0 | |
| 16 | Rsvd | 0 | |
| 15 | Rsvd | 0 | |
| 14 | Rsvd | 0 | |
| 13 | Comet D Int | Comet D Int | 1 = asserted |
| 12 | Comet C Int | Comet C Int | 1 = asserted |
| 11 | Comet B Int | Comet B Int | 1 = asserted |
| 10 | Comet A Int | Comet A Int | 1 = asserted |
| 9:0 | Rsvd | X | |

### 5.3.4 Status Register 2 (0x2010)

This 32-bit register is used to read interrupt and FPGA configuration status.

**Table 24: Status Register**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 31:22 | Rsvd | 0 | |
| 21:20 | Rsvd | PLD Ctrl | |
| 19:18 | Rsvd | PLD Status | |
| 17:16 | Rsvd | PLD Jumper | |
| 15:0 | Rsvd | PLD Spares | |

## 5.4 Bank 3 Registers (DPT5/6 only)

The Bank 3 Registers are 8 bits wide and start in Bar 0 at offset 0x3000 as follows:

**Table 5.3: Bank 3 Registers**

| Offset | 0 | 4 | 8 | C |
|---|---|---|---|---|
| 0x00 | Ctrl 0 | Ctrl 1 | | |
| 0x10 | | | | |
| 0x20 | Status | | | |
| 0x30 | | | Rev Code MSB | Rev Code LSB |
| 0x40 | | | | |
| 0x50 | | | | |
| 0x60 | | | | |
| 0x70 | | | | |
| 0x80 | | | | |
| 0x90 | | | | |
| 0xA0 | | | | |
| 0xB0 | | | | |
| 0xC0 | | | | |
| 0xD0 | | | | |
| 0xE0 | | | | |
| 0xF0 | | | | |

### 5.4.1 Control Register 0 (0x3000)

This 32-bit register is used to control a variety of functions.

**Table 22: Control Register 0**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 7:1 | Rsvd | 0 | |
| 0 | LED | LED | |

Notes:

### 5.4.2 Control Register 1 (0x3004)

This 32-bit register is used to control a variety of functions.

**Table 22: Control Register 0**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 7:0 | Rsvd | 0 | |

Notes:

### 5.4.3 Status Register (0x3020)

This 32-bit register is used to control a variety of functions.

**Table 22: Control Register 0**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 7:4 | Rsvd | Cfg State | |
| 3 | Reserved | 0 | |
| 2 | Reserved | Error | |
| 1:0 | Reserved | PLD Jumpers | |

Notes:

### 5.4.4 Rev Code MSB (0x3038) and LSB (0x303C)

This 32-bit register is used to control a variety of functions.

**Table 22: Control Register 0**

| Bit(s) | Write | Read | Notes |
|---|---|---|---|
| 7:0 | Rsvd | Rev Code (MSB or LSB) | |

Notes:

# 6. Channelization

One of the features of the DPT5/6 is the ability to perform channelization on the E1/T1 data streams. Channelization is the process of making a translation between the fixed time-ordering of the E1/T1 data and the logical ordering of the data in the user buffers, such that the data is more directly usable by the user application, thus taking this processing burden from the user. In the DPT5/6, this function is performed in hardware. This document describes in detail how the DPT5/6 channelization operates, and what the necessary interfaces are at the embedded and host levels.

## 6.1 Operation Overview

The channelizer hardware is implemented in the Host FPGA and works in conjunction with four E1 framers and an internal 128Kbyte DPRAM, where the channelized data is stored. The channelizer has access to one side of the DPRAM, while the Chronus processor has access to the other side. The channelizer uses and maintains two sets of tables (also located in the DPRAM) to determine where to store received E1/T1 data and from where to retrieve E1/T1 transmit data. When the receive data buffers are full, or the transmit buffers are empty, the channelizer interrupts the Chronus. The Chronus then transfers data between the smaller DPRAM buffers and larger buffers in the local memory (16Mbytes of SDRAM). Again, when these local memory receive data buffers are full, or transmit buffers are empty, the Chronus interrupts the host and coordinates transfers between the DPT5/6 local memory and host buffers via the 32bit/33Mhz host PCI bus.

## 6.2 Operation Details

Figure 3 shows a block diagram of the DPT5/6 channelizer subsystem. The DPT5/6 has four bidirectional E1 or T1 interfaces. For clarity, the receive path will be described first. Each of the four E1/T1 framers provide three serial receive backplane signals: clock, data and frame. The BRIF blocks use these to convert the serial data in to 8-bit timeslot data. Each time a new 8-bit timeslot is received, the BRIF signals to the channelizer that a new slot data byte is available.

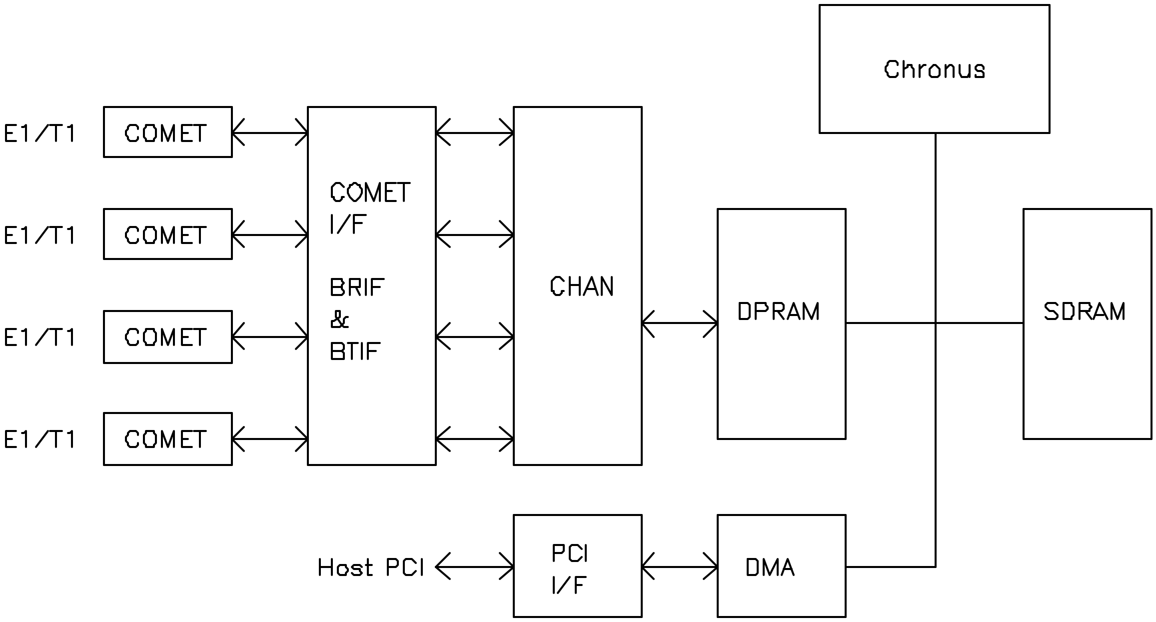


Figure 3: Channelization Block Diagram

At this point the channelizer must write the new data into the DPRAM at the appropriate location. To understand this, the DPRAM is now described.

### 6.2.1 DPRAM Organization

The Figure below shows the DPRAM organization. The DPRAM is organized as two 64Kbyte banks. This is to allow the channelizer and the Chronus to each access the DPRAM at the same time without any risk of collisions. Each bank has three sections – two look-up tables, and the data buffer area. The tables are initialized by the user and determine how the data is channelized.

| | | |
|---|---|---|
| Bank 0 | 0x00000 | Channel Table 0 |
| | 0x00100 | Pointer Table 0 |
| | 0x00200 | Data Area 0 |
| | 0x0FFFF | End 0 |
| Bank 1 | 0x10000 | Channel Table 1 |
| | 0x10100 | Pointer Table 1 |
| | 0x10200 | Data Area 1 |
| | 0x1FFFF | End 1 |

Figure 4: Channelization DPRAM Organization

### 6.2.2 Channel Table

Table is the Channel Table. This is a 256 x 16-bit table where the index is an 8-bit value formed by concatenating one bit indicating the direction (0 for Rx, 1 for Tx), two bits indicating the framer (0-3) and five bits indicating the slot (0-31). For example, if a byte is received (direction is “0”) on framer 2 (10b), slot 7(00111b), the index would be (in binary) {0,10,00111} = 0x47. The Channel Table value (at index 0x47 in this case) is the channel identifier for that direction/framer/slot.

**Table 25: Channel Table**

| Table Offset | Direction | Framer |
|---|---|---|
| 0x00-0x1F | Receive | 0 |
| 0x20-0x3F | Receive | 1 |
| 0x40-0x5F | Receive | 2 |
| 0x60-0x7F | Receive | 3 |
| 0x80-0x9F | Transmit | 0 |
| 0xA0-0xBF | Transmit | 1 |
| 0xC0-0xDF | Transmit | 2 |
| 0xE0-0xFF | Transmit | 3 |

The channel identifier determines how slots are grouped into buffers. For example, if the receive time slots 0, 5 and 12 of framer 3 are all given the same channel identifier N, the data for those slots will be stored sequentially in the buffer for that channel as shown here:

Channel N Buffer: …, d($F_nS_0$), d($F_nS_5$), d($F_nS_{12}$), d($F_{n+1}S_0$), d($F_{n+1}S_5$), d($F_{n+1}S_{12}$), … etc.

where d($F_iS_j$) is the time slot data for frame i, slot j.

The channel identifier is a 16-bit value. The lower byte is similar to the channel index, where the upper three bits indicate the direction (0,1) and framer (0-3). The low 5 bits are the channel number (0-31), and identifies a group of one of more time slots for that particular direction and framer. *Note that channels can only include slots from one direction/framer.*

The upper bit of the upper byte serves as the VALID bit in the receive direction, and as the IDLE bit in the transmit direction (transmit operation is discussed later). If VALID is 1, the received data is stored in the channel buffer per the tables. If VALID is 0, this indicates that this slot is not part of any channel and the received data is discarded. Thus, filling the Channel Table with 0x0000 will clear all channel assignments. Here is a summary of the channel identifier (x = don’t care) in binary:

| | |
|---|---|
| 0xxxxxxx xxxxxxxx: | Invalid/unassigned channel – data discarded |
| 1xxxxxxx 0ffccccc: | Valid channel – data from framer ‘ff’ is stored in the channel ‘ccccc’ buffer |

### 6.2.3 Pointer Table

Table 26 is the Pointer Table. This is also a 256 x 16-bit table where the index is the lower byte of the Channel Table value (direction/slot/channel). The Pointer Table value is a 16-bit byte offset into the DPRAM. Note that to fully address the 128Kbyte DPRAM space, 17 address bits are required. In this case the upper bit comes from the bank select, which is controlled by the channelizer hardware. The Pointer Table value provides the other 16 bits of address.

**Table 26: Pointer Table**

| Table Offset | Direction | Framer |
|---|---|---|
| 0x00-0x1F | Receive | 0 |
| 0x20-0x3F | Receive | 1 |
| 0x40-0x5F | Receive | 2 |
| 0x60-0x7F | Receive | 3 |
| 0x80-0x9F | Transmit | 0 |
| 0xA0-0xBF | Transmit | 1 |
| 0xC0-0xDF | Transmit | 2 |
| 0xE0-0xFF | Transmit | 3 |

Continuing the previous example, when a byte is received (direction is “0”) on framer 2 (10b), slot 7(00111b), the index would be (in binary) {0,10,00111} = 0x47. Suppose the Channel Table value at index 0x47 is 0x40 (channel 0 for this direction/framer). The channelizer will read the Pointer Table at index 0x40 to get a 16-bit address. This address points to the location in the active DPRAM bank that the received byte will be written. The channelizer will write the data to this location, increment the pointer, and then write the new pointer back to index 0x40. Thus the next data received that is part of this channel will be stored in the next byte address of that channel’s buffer.

The Pointer Table must be initialized according to the Data Buffer mapping, described below. The initialization of the Pointer Table is dependent on a number of user application-specific parameters, such as the number of channels, the number of slots per channel, and the latency of the system.

Once initialized, all operations associated with these tables and the movement of data are performed in hardware. No user intervention is required other than that of managing buffers as they become full/empty.

### 6.2.4 Data Buffers

The Data Buffer area in each bank is 32Kbytes minus the size of the tables (1K) which totals 31Kbytes (31744 bytes). The user must allocate this space among the channels expected to be open. In general, the maximum buffer size for a given channel would be

(31744 / (the total number of total slots used in all channels)) * (slots in this channel).

This size must also be a multiple of 4. For example, if all 32 slots from all four framers are used in channels, and the channel in question includes 5 slots, the maximum size for that channel buffer would be

(31744/(32*4)) * 5 = 1240 bytes.

In some applications, such as where low latency is required, a smaller size may be required. The size of the buffer determines how often the Chronus will be required to move data between the DPRAM and the SDRAM.

Note that although there may be multiple channels of different sizes (numbers of slots in them) for a given framer, the buffers will all fill at the same rate, and all be full at the same time. However, if channels are created for different framers, and the incoming E1/T1's are not synchronous, channels from one framer may fill at a slightly different rate than those from another framer, thus becoming full at different times, and requiring that the Chronus manage buffers from separate framers independently.

### 6.2.5 Receive Controls

The Rx Control register in the Host FPGA is used to control channelization in the receive direction (see section 5.3.5).

Typically, all channelization initializations are done with the Rx Enables (bits 3-0) cleared. This includes the framers (framer initializations are not discussed in this document), the DPRAM tables and buffers, the IRQ Mask Register, and writable bits of this register. The "Frames Per Buffer – 1" bits should be set to

(31744 / (the total number of total slots used in all channels)) - 1

up to a maximum of 255 (use 0 for 256). Clear any IRQ's before starting.

Once running, this register is used to clear IRQ's, and to check bank and overflow status.

### 6.2.6 Transmit & Controls

In general, the channelization operation for transmit is similar to that of receive, except in reverse. See Figure 1. The BTIF blocks in the Host FPGA must take 8-bit timeslot data from the channelizer, and then serialize it. In the DPT5/6, all 4 transmitters are synchronized to each other, based on a user-selected timing source. Each time the transmitter BTIF blocks need another byte, they signal the channelizer, indicating which slot it needs. The channelizer uses the direction (1 for Tx), framer (0-3) and slot (0-31) to form the index into the Channel Table, and the Channel Table value is used to index into the Pointer Table. Data is read from the DPRAM buffer at the given pointer, and sent to the BTIF and on out to the framers. The Tx Control register in the Host FPGA is used to control channelization in the transmit direction (see section 5.3.6 ).

## 7. E1 vs. T1

The DPT5/6 can be run in either E1 or T1 mode through software selection. All four framers must be in the same mode. The following sections discuss the unique aspects of either mode.

### 7.1 E1

In E1 mode, the standard receive input impedance is set to 120 ohms. The DPT5/6 can be optionally populated for 75 ohm receive input impedance. This option simply adds a resistor (45.3 ohms) to each of the framer input circuits (R25, R36, R37, R58).

The bit rate for E1 is 2.048 MHz, and all related clocks run at this rate, as shown in this table:

Table 27: E1 Bit Rate

| Signal | Rate | Notes |
|---|---|---|
| BRCLK | 2.048 MHz | Receive Backplane Clock |
| BTCLK | 2.048 MHz | Transmit Backplane Clock |
| TXCLK | 2.048 MHz | Transmit Clock |

### 7.2 T1

In T1 mode, the standard receive input impedance is set to 100 ohms.

Although the bit rate for T1 is 1.544 MHz, not all clocks run at this rate, as shown in this table:

Table 28: T1 Bit Rate

| Signal | Rate | Notes |
|---|---|---|
| BRCLK | 1.544 MHz | Receive Backplane Clock |
| BTCLK | 2.048 MHz | Transmit Backplane Clock |
| TXCLK | 2.048 MHz | Transmit Clock |

When the framer is properly set up for T1, it handles the conversion from 2.048 MHz to 1.544 MHz.

# 8. DMA

In a typical system, the data transfer rate for non-DMA transfers _to_ the DPT5/6 are much higher than that of non-DMA transfers _from_ the DPT5/6. Therefore, the DPT5/6 provides the ability to allow DMA for data transfers from DPT5/6 (SDRAM) to host. This section describes operation of this DMA.

The Chronus initiates the DMA. It must first get the host memory address for the DMA destination. Two Host FPGA registers are used to set up a DMA. The DMA Word Count register sets the number of 32-bit words to be transferred, and the DMA Host Address register sets the destination address for the DMA. When the DMA Host Address register has been written, the DMA is armed. At this point, the next 'modified' SDRAM read will not only start the DMA, but also sets the SDRAM source address. The SDRAM read which starts the DMA must be to a 'modified' SDRAM address, where the address is a normal SDRAM address except that bit 27 of the address must be set (it is normally 0). DMA will not occur unless this is the case.

Three DMA status bits (the 3 LSB's of the Hardware Control register) are available for further control. The DMA FIFO Not Full bit indicates whether it is all right to start the DMA (if it is low, wait, else proceed). The DMA Armed bit indicates that the DMA Host Address register has been written and the DMA controller is waiting for the DMA to be started with an SDRAM read (with A27=1). When that read occurs, the DMA Armed bit will clear, and the DMA Busy bit will assert. When the DMA is done, the DMA Busy bit will clear.

Here is a possible DMA algorithm:

(1) Verify that DMA Armed and DMA Busy bits are cleared
(2) Write DMA Word Count
(3) Read DMA FIFO Not Full bit. If 1, proceed
(4) Write DMA Host Address
(5) Verify that DMA Armed bit is set (optional)
(6) Read SDRAM
(7) Verify that DMA Armed bit is cleared (optional)
(8) Verify that DMA Busy bit is set (optional)
(9) Wait for DMA Busy bit to be cleared
(10) Repeat

# 9. TDM

The TDM subsystem allows serial bus interconnection between the E1/T1 interface and the two smPCI module sites. The TDM subsystem has these fixed parameters:

- 8.192 MHz TDM clock rate
- 8 KHz TDM frame rate
- 8 bits per TDM time slot
- 128 TDM time slots per TDM frame

## 9.1 TDM Signals

The TDM subsystem uses these signals:

**Table 29: TDM Signals**

| Group | Signal | Function |
|---|---|---|
| Timing | TDMCLK | 8.192 MHz TDM timing base |
| | TDMSLOT | 1.024 MHz (1/8 duty cycle), one pulse per time slot |
| | TDMFRAME | 8 KHz (1/256 duty cycle) frame pulse |
| | TDMSF | User-programmable, high for one frame low for others |
| Control | TDMCTL[7:0] | 8-bit control word, updated every TDMCLK |
| Data | TDMD[3:0] | Four Bi-directional serial data buses that interconnect smPCI modules to TDM |
| | TDMV[3:0] | Four Bi-directional serial buses that interconnect SmPCI modules to TDM, typically used to indicate validity of the corresponding TDMD bus. |

## 9.2 TDM Control

A 2048 x 8 bit dual-port memory is used to control TDM data/valid bus activity. This memory is set up as two banks of 1024 x 8 bits. One bank is accessible by the user, while the other is used by the TDM hardware. The TDM hardware reads out this memory sequentially and presents it onto the TDMCTL bus such that a new TDMCTL word appears for each TDMCLK cycle. Thus, the TDMCTL words at locations 0-7 control the first TDM time slot, locations 8-15 control, the second slot, etc. Since each word is 8 bits, a total of 64 bits can be used to control each time slot. TDMCTL word mapping is as follows:

**Table 30: TDM Control Words**

| TDMCTL Word | MS Nibble | LS Nibble |
|---|---|---|
| 0 | TDM0 Source Control | TDM1 Source Control |
| 1 | TDM2 Source Control | TDM3 Source Control |
| 2 | Reserved | Reserved |
| 3 | Reserved | Reserved |
| 4 | smPCI Module A Ctrl | smPCI Module B Ctrl |
| 5 | Reserved | Reserved |
| 6 | E1/T1 Tx Source Ctrl | Reserved |
| 7 | smPCI Module A Dest Ctrl | smPCI Module B Dest Ctrl |

### 9.2.1 TDM Source Controls

TDM Source Controls are defined as follows:

**Table 31: TDM Source Control**

| Control Nibble | TDM driven by |
|---|---|
| 0000 | smPCI Module A |
| 0001 | smPCI Module B |
| 001x | Reserved |
| 010x | Reserved |
| 0110 | smPCI Module A |
| 0111 | smPCI Module B |
| 1000 | E1/T1 Rx Data |
| 1001 | E1/T1 Rx Status |
| 1010 | Reserved |
| 1011 | Reserved |
| 1100 | Source Continuation |
| 1101 | Reserved |
| 1110 | Reserved |

| | |
|---|---|
| 1111 | None |

### 9.2.2 E1/T1 Transmit Source Controls

E1/T1 Tx Source Controls are defined as follows:

**Table 33: E1/T1 Tx Source Control**

| Control Nibble | E1/T1 Tx Driven By |
|---|---|
| 0000 | TDM0 |
| 0001 | TDM1 |
| 0010 | TDM2 |
| 0011 | TDM3 |
| 0100 | E1/T1 Rx Data |
| 0101 | E1/T1 Rx Status |
| 011X | 0 |
| 1XXX | 0 |

## 9.3 TDM Timing Diagram

The following shows relative functional timing for the TDM subsystem.

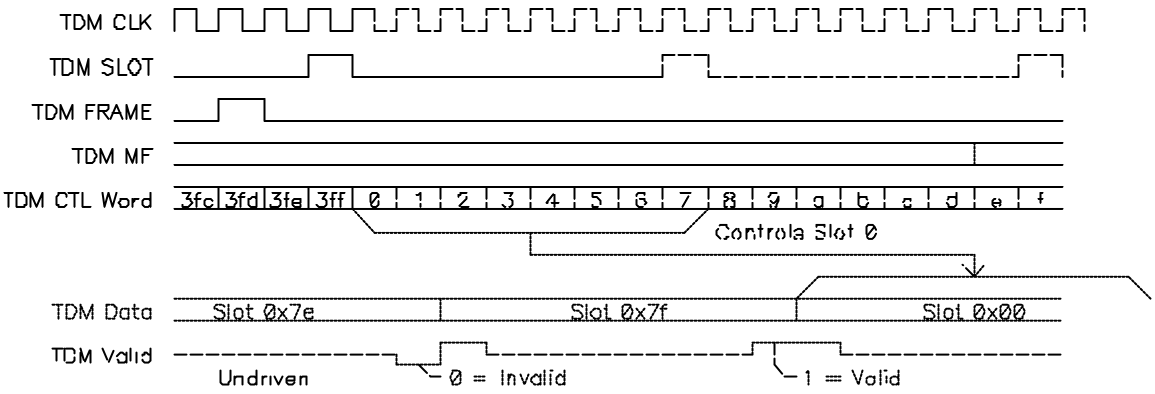


**Figure 5: TDM Relative Functional Timing**

## 9.4 TDM Timing Sources

TDM timing may be programmed to be based on one of three sources:

- Local Crystal
- Fastest Recovered E1/T1 Receive Clock

### 9.4.1 Local Crystal Mode

In this mode, a local crystal-based clock synthesizer provides the timing for the TDM subsystem.

### 9.4.2 Fast Receive Clock Mode

In this mode, the hardware dynamically determines the fastest E1/T1 recovered receive clock and automatically slaves the TDM timing to it. The user can select which E1/T1 ports are to be candidates for the “fast clock” logic. This mode is useful in applications where the DPT5/6 is receiving E1/T1 data from multiple time bases. In general, in this case, frame slips will occur, and those slips may be either lost frames or repeated frames. However, in this mode, since the fastest one is selected, all slips will be repeated frames, and therefore no data will be lost. Repeated frames are marked as “invalid” in the TDM subsystem.

## 10. H.100

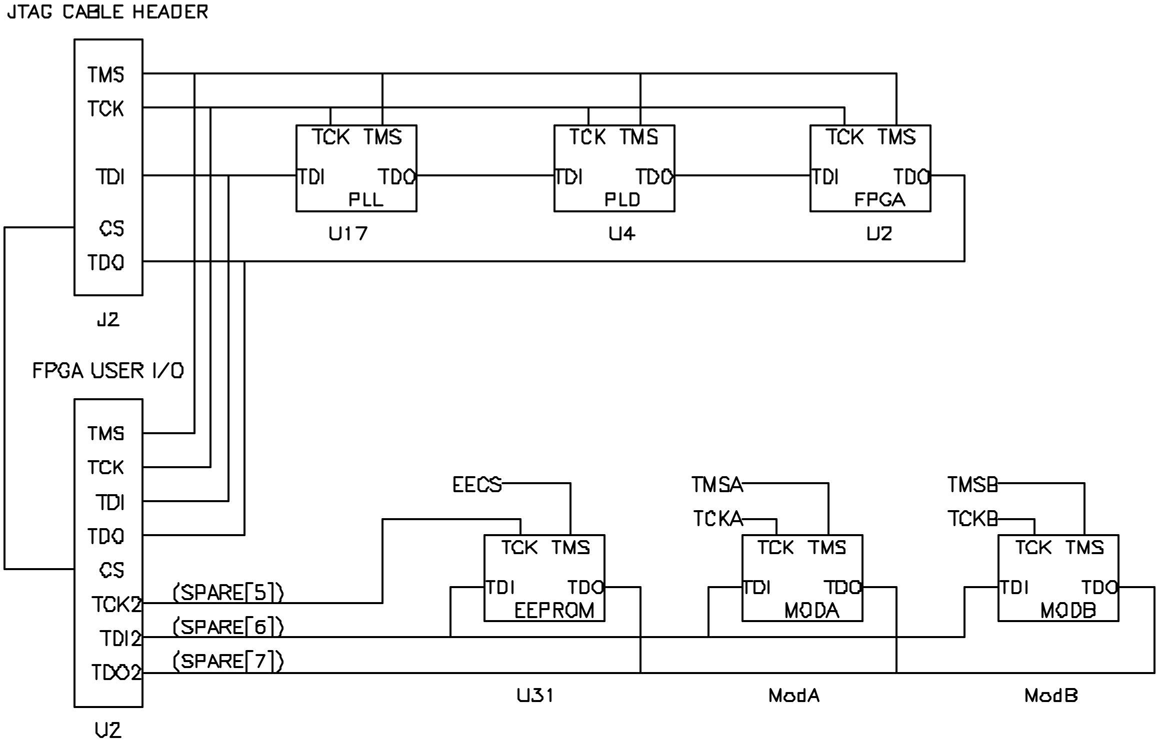
The DPT5/6 does not include an H.100 interface.

## 11. JTAG

Two JTAG chains exist on the DPT5/6 for access to the following devices

- PLL
- PLD
- FPGA
- EEPROM
- SmPCI A
- SmPCI B

Here is a diagram of the JTAG configuration:



The JTAG Cable Header is use in manufacturing to configure the PLL, PLD and FPGA, and is also used for FPGA debug. The FPGA JTAG port is used with the provided API to configure any of the devices in-system for any required updates or configuration changes.

## 12. Serial EEPROM

The DPT5/6 has a 64 x 16-bit serial EEPROM device which is accessed through the JTAG Control register. The first 16 words are reserved for OEM use, and the remaining 48 words are available to the user for general use. The organization is as follows:

**Table 35: EEPROM Organization**

| Location | Function |
|---|---|
| 0 | Chronus Clock Freq |
| 1 | E1/T1 Mux Clock Freq |
| 2 | Local PCI Clock Freq |
| 3 | DPT5/6 Options |
| 4 | PLD Version |
| 5 | Host FPGA Version |
| 6 | Expansion FPGA Version |
| 7 | MIPS Loader Version |
| 8 | Magic Number |
| 9 | MIPS Application Version |
| A | Reserved |
| B | Reserved |
| C | Serialization Date |
| D | Board Revision |
| E | Board Serial Number |
| F | Burn in Hours |
| 0x10-0x3F | User Space |

## 13. Power-up Sequence

The DPT5/6 FPGA must be configured before Host or Chronus applications access the board resources. The PLD works in conjunction with a flash memory to perform this task. Note that the PLD and flash must be appropriately configured for this to work.

When the board is powered on, the PLD monitors the FPGA configuration status signals. It takes control of the Chronus bus to the extent that it can sequentially read the FPGA configuration data out of the flash and write it to the FPGAs. When the PLD sees that the FPGA is configured it relinquishes the Chronus bus.

## 14. Configuring a New DPT5/6

As discussed above, the DPT5/6 requires that its programmable devices be initialized. In order to do so, a specific sequence must be followed.

### 14.1 Using a Hot Swap Extender

In this sequence, it is assumed that the board is in a hot swap extender, and that the host was booted with a configured board prior to this sequence. The board name is assumed to be dpt0.

(1) Boot the system with a known good DPT5/6 in the extender and confirm the the DPT5/6 is operational

(2) Turn off the extender and install a new DPT5/6

(3) Attach the Altera ByteBlaster USB pod to J2 and turn on the extender.

(4) Configure the PLL, PLD and FPGA (see separate instructions)

(5) Run “dpclock dpt0”. This reconfigures the PLL.

(6) Run “dpflashup dpt0”. This reconfigures the PLD and FPGA, and loads MIPS code.

(7) Cycle extender power to force self-configuration as described above.

### 14.2 Without a Hot Swap Extender

In this sequence, it is assumed that the board is not in a hot swap extender. The board name is assumed to be pci0.

(1) TBD

# 15. Interfaces

This section describes the connectors, LEDs and jumpers on the board.

## 15.1 Connectors

### 15.1.1 PCI/PCI-X Bus Interface

The bottom card edge of the board provides the Universal 32-bit Short PCI Bus interface per PCI Rev. 2.1 Specification, sections 4.3.7 and 5.2.

### 15.1.2 E1/T1 Line Interface

The E1/T1 line interfaces use EMI Shielded RJ45 connectors. The RJ45 connector pinout is shown here:

| Pin | Signal |
|---|---|
| 1 | Rx Ring |
| 2 | Rx Tip |
| 3 | No Connection |
| 4 | Tx Ring |
| 5 | Tx Tip |
| 6 | No Connection |
| 7 | No Connection |
| 8 | No Connection |

## 15.2 LEDs

### 15.2.1 E1/T1

There are 4 red user-programmable LED's behind the E1/T1 connectors. These may be used to indicate line activity, frame sync, etc.

### 15.2.2 Core

On the top edge of the card, there are two red/green surface mount LED's. The left one is labeled PLD and the right one is labeled FPGA. Both used as follows.:

| | |
|---|---|
| Red: | device is not configured |
| Off: | device is configured, and is driving the green LED off |
| Green: | device is configured, and is driving the green LED on |

### 15.2.3 Expansion

On the top edge of the card, above the smPCI connectors, there are two red/green surface mount LED's. The left one is labeled SMPCIA and the right one is labeled SMPCIB. Both used as follows.:

| | |
|---|---|
| Red: | smPCI module is not configured |
| Off: | smPCI module is configured, and is driving the green LED off |
| Green: | smPCI module is configured, and is driving the green LED on |

## 15.3 Jumpers

### 15.3.1 JP2

This 2x2 jumper block is above the smPCI B site and is used to control how the PLD configures the FPGA at power-up. With no jumpers, the PLD loads the FPGA with the 'current' FPGA configuration from Bank 1 of the flash. If either jumper is installed (jumpers are installed horizontally), the PLD loads the FPGA with the 'backup' FPGA configuration from Bank 0 of the flash.

### 15.3.2 Write Enable Jumpers

This 2x2 jumper block is above the smPCI B site and is used to enable writing to the flash or EEPROM memories. The top jumper (jumpers are installed horizontally), is labeled "EE WR EN" and enables writing to the EEPROM when installed. The bottom jumper is labeled "FL WR EN" and enables writing to the flash when installed.

## 16. Mechanical Characteristics

The DPT5 is a Universal 32-bit Short PCI Bus card, as described in the PCI Rev. 2.1 Specification, section 5.2. The overall dimension of the board is 6.6" long, 4.2" wide. The DPT6 is an X1 Short PCIe Bus card, as described in the PCI Express Card Electromechanical Specification, Rev. 1.1, section 6. The overall dimension of the board is 6.6" long, 4.376" wide.

## 17. Power Consumption

The DPT5/6 uses the +5V power from the PCI Bus. Typical power consumption is 4.0 Watts with no smPCI Modules installed. Maximum power consumption is estimated to be less than 7.5 Watts with no smPCI Modules installed. Additional power due to smPCI Modules may be derived from specifications on the installed modules.

## 18. Software

Software drivers for Solaris are available. See the DPT5/6 Programmer Reference Manual for information at www.cacdsp.com. Windows NT drivers are to be provided, together with a host API to allow control of the DPT5/6 from user-developed programs.

Diagnostics are provided which are sufficient to verify correct operation of all major functions of the DPT5/6.

## 19. Sources Consulted

1. PCI Special Interest Group. The PCI Local Bus Specification, Revision 2.1. June 1, 1995.