TITLE :

ASIC/FPGA Technical Manual

PEV-1100 HW/SW Interface Cassidian MAS Technical Reference

KEY WORD :

ISO9000, Quality System, IOxOS Quality System, FPGA, ASIC

ACCESS CODE :

CONFIDENTIAL

SUMMARY :

This document provides the PEV_1100 "PCI_Express to VME64X Bridge Hardware-Software Interface" and the "Hardware Technical Reference"

IOxOS Technologies SA
4, Chemin des Fontenailles
CH-1196 Gland
SWITZERLAND

PROJECT REFERENCES

Internal project, External contact, ...

DOCUMENT REFERENCE No:	*INTERNAL_1 :*	*INTERNAL_2 :*

	Execution	*Date*
AUTHOR	[JB] J. Bovier	25 December 2008
CHECKED	[JFG] J. F. Gilot	25 December 2008
APPROVED		
AUTHORIZED		

REVISION	DATE	CHAPTER AND SECTION MODIFICATIONS	APPROVED	AUTHORIZED
1.0	01/09/11	Add CASSIDIAN specific requirements	J. Bovier	

Author references

INITIAL	NAME	COMPANY	COMMENTS
[JB]	Joel BOVIER	IOxOS Technologies SA	
[JFG]	Jean-François GILOT	IOxOS Technologies SA	
...			

20

Table of Contents

Index of Tables

Index of Drawings

25

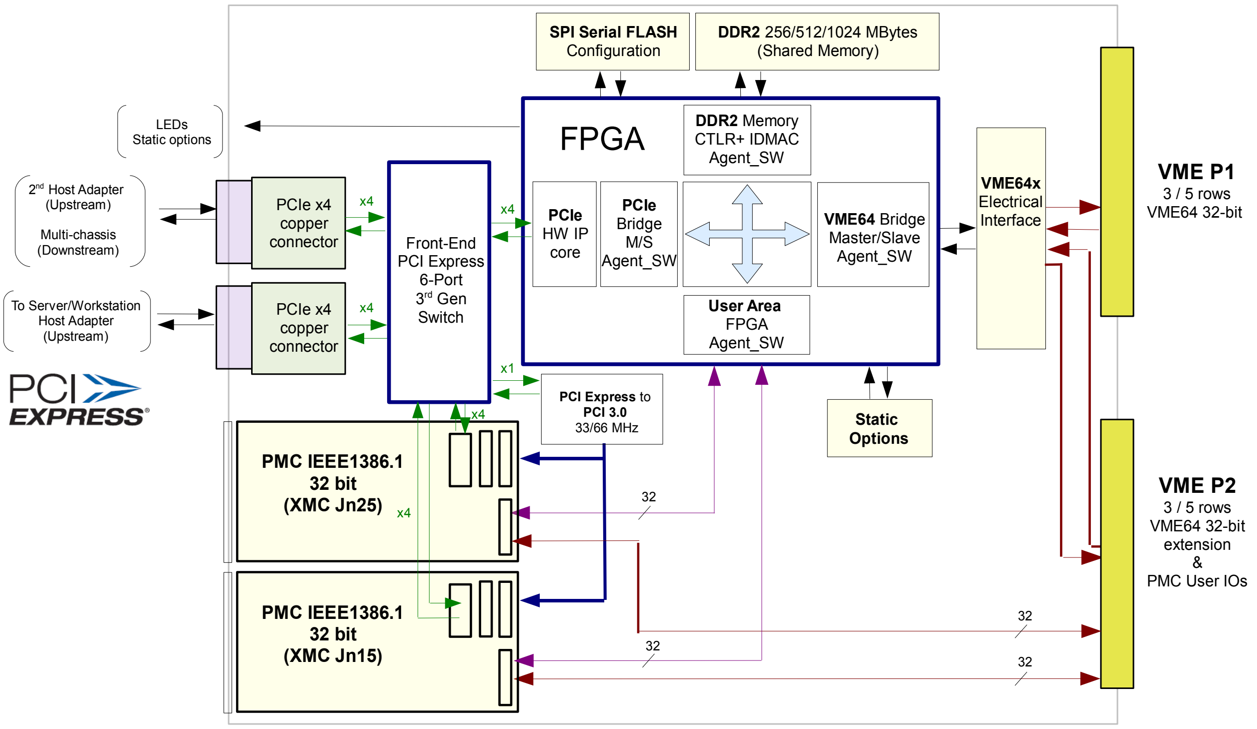
1 Introduction

The PEV_1100 is designed following the rules and documentation procedures defined by the “IOxOS Technology SA Quality System”.

1.1 PEV_1100 Project Overview

The PEV_1100 is built on the latest PCISIG “PCI Express External Cabling Specification 1.0” allowing to extend the PCI Express Bus infrastructure over medium distance (up to 7 meters) with high speed external cabling. It is defined to support PCI Express from x1 up to x16 lanes. The connectors and the cables are fully specified by the PCISIG and widely available on the market.

In a first stage, the PCI Express x4 Host Adapter Card, widely available by 2nd third party supplier can be used for the server/workstation connection. In a second step, PCI Express External Cabling connectors will be directly available on the PC ATX/ATXE mother-board. For laptop connectivity, ExpressCard plug-in provides the PCI Express x1/x4 cable connection.



Drawing 1: PEV_1100 Block Diagram

The PEV_1100 supports multi-chassis expansion through its 2nd PCI Express x4 connector when configured as DOWN_stream. This 2nd connector can also be configured as UP_stream enabling Dual-Host / Dual-Fabric system configurations.

For industrial applications, 19” rack mount equipment linked with the PEV_1100 delivers a cost-effective and reliable solution. All major IT suppliers (IBM, Sun, Dell and HP among others) provide high performance 19” rack mount equipment for servers/workstations in 1U / 2U form factors at a very competitive price (a fraction of the cost of VME SBC).

A PEV_1100 based system can be extended up to 16 chassis when using a PCI Express Switch (available in 1U form factor).

- VME64x 6U Board with two XMC/PMC sites
  - VME64x with 3 / 5 rows connectors
  - 5[V] only power supply
- Transparent PCI Express to VME64x Interface
  - Multi-chassis capability (IN-OUT chaining)
  - Dual-Host Support
  - Full Interrupt Dispatching
- PCI Express External Cabling Specification Rev 1.0
  - Two PCIe x4 External Connectors
  - PCIe x4 copper cable (up to 7 meters)
  - Optional SFP Optical Adapter (up to 100 meters)
- On-board 256/512/1024 MBytes DDR2 Memory
  - Multi-port shared access
  - High bandwidth R/W (2 GBytes/s)
- Embedded 4-channel Intelligent DMA Controller (IDMAC)
  - Optimized for PCI Express architecture
  - Any Source/Destination port (VME64x, both PMC/XMC sites, PCI Express and Shared Memory)
  - Chaining support
- Two PMC IEEE 1386.1 / XMC VITA 42.3 sites
  - PCI 3.0 32-bit @ 33/66 MHz
  - Direct bridge from PCI Express
  - XMC VITA 42.3
- VME64 Master
  - CR/CSR, A16, A24, A32 and user defined
  - SBT, BLT32, MBLT64, 2eVME & 2eSST (VME320)
  - Full VME Slot-1 Function
- VME64 Slave
  - Local Shared Memory
  - Direct Bridge to PCI Express & PMC/XMC
  - Direct Bridge to PCI Express UP_stream
  - SBT, BLT32, MBLT64, 2eVME & 2eSST (VME320)
- Standard (0ºC to 70ºC) and Extended (-40ºC to 85ºC)
- Built on the latest FPGA technology with PCI Express embedded support
  - Full control on VHDL source code
  - Obsolescence management
  - User area available for custom applications
  - FPGA Design Kit
- API & Libraries available for Window XP and Linux

Applications

- Enhanced & High performance VME64x Single Board Computers (SBC) and IO control
- Aerospace integration rig systems, flight simulators & test equipment

50

1.2 Acronyms and Abbreviation.

Following table sums up specific abbreviations and acronyms used in this document.

ABBREVIATION	MEANING
CFG	Constant Frequency Clocking. PCI Express clocking mode.
CPLD	Complex programmable Logic Device
ITC16	Interrupt Controller
ITP	Interrupt pending
PCI Express	Serial based PCI Interfacing defined by PCISIG Organization.
VME64X	VME 64-bit, defined by ANSI standard and VITA organization

ABBREVIATION	MEANING
FPGA	Field Programmable Gate Array
PMC (IEEE1386.1)	PCI Mezzanine Card, referenced as PMC, normalized under IEEE1386.1
SSC	Spread Spectrum Clocking. PCI Express clocking mode.
XMC	Extended Mezzanine Card, VITA 42 specification
PrPMC	Processor PMC Mezzanine, VITA32
XILINX	FPGA Manufacturer

Table 1.1 : List of Acronyms and Abbreviation

1.3 Related and Referenced Document.

Following table sums up the main document references. (IOxOS proprietary, Specifications, Technical documents, customer related, …)

Additional informations are also available to the related manufacturer.

REF	TITLE	DOC IDENTIFICATION	SOURCE
{1}	ASIC/FPGA Design Process STD	Q_PROC_AFD_0001_An	IOxOS Quality System
{2}	VME64 Standard	VITA 1-1994 (R2002)	VITA
{3}	VME64 Extension VME64X 2eSST Protocol	VITA 1.1-1997(R2005) VITA 1.5-1997(R2005)	VITA
{4}	PCI Express Specification 1.1A PCI Express Specification 2.0	PCISIG	PCISIG
{5}	PCI Express External Cabling Specification 1.0	PCISIG	PCISIG
{6}	Conduction Cooled PMC	VITA 20-2001(R2005)	VITA
{7}	Processor PMC (PrPMC)	VITA 32-2003	VITA
{8}	PMC-P4 Pin Out Mapping To VME-P2	VITA 35-2000	VITA
{9}	Status Indicator Standard	VITA 40-2003	VITA
{10}	VXS VME Switched Serial Standard	VITA 41.0-2006, 41.2-2006,	
{11}	XMC Switched Mezzanine Card (PCI Express)	VITA 42.0-200x, 42.3-2006	VITA
{12}	PCI Local Bus Specification 3.0	PCISIG	PCISIG
{13}	Xilinx Virtex-5 family data sheet	DS100.pdf	XILINX
{14}	Xilinx Virtex-5 FPGA User Guide	UG190.pDF	XILINX
{15}	Xilinx Virtex-5 FPGA System monitor User Guide	UG192.pdf	XILINX
{16}	Xilinx Virtex-5 FPGA Integrated PCI Express End Point.	UG197.pdf	XILINX
{17}	Xilinx Virtex-5 LogiCORE Endpoint Block for PCI Express Designs	UG350.pdf	XILINX
{18}	PLX Technology PEX-8624-AA	Express Lane PEX-8624-AA 24-lane/6-port PCI Express Gen 2 Switch Data Book	PLX Technology
{19}	PLX Technology PEX-8112-AA	PEX_8112-AA_Data Book PCI Express x1 to PCI Bridge	PLX Technology
{20}	SPI Serial Flash M25P128	Data-sheet M25P128	Numonix

Table 1.2 : Related and Referenced Documents

1.4 Environmental

The PEV_1100 is designed for commercial and industrial temperature grade. The above table sums up the environmental.

	PEV_1100 Commercial	PEV_1100 Industrial	Comments
Operational Temperature	0 to 60 [ºC]	- 40 to 75 [ºC]	With 400 LFM forced air cooling.
Storage Temperature	- 55 to 105 [ºC]	- 55 to 105 [ºC]	
Relative Humidity	5% to 95%	5% to 95%	Non condensing
Vibration	To Be Defined	To Be Defined	
Shocks	To Be Defined	To Be Defined	
Altitude	0 – 5000 meters	0 – 5000 meters	
EMI / RFI	EN50022-Class A EN50082	EN50022-Class A EN50082	
Security	EN60950	EN60950	

Table 1.3.: PEV_1100 Environmental

Due to the difficulties to get precise informations on the air cooling efficiency and the installed PMC/XMC, the PEV_1100 provides on-board thermal sensors.

- Internal XILINX Virtex-5T die temperature monitor. Refer to chapter 2.5.3
- External LM86 temperature sensor.

These two thermal informations allow to characterize the thermal envelope for a PEV_1100 specific integration.

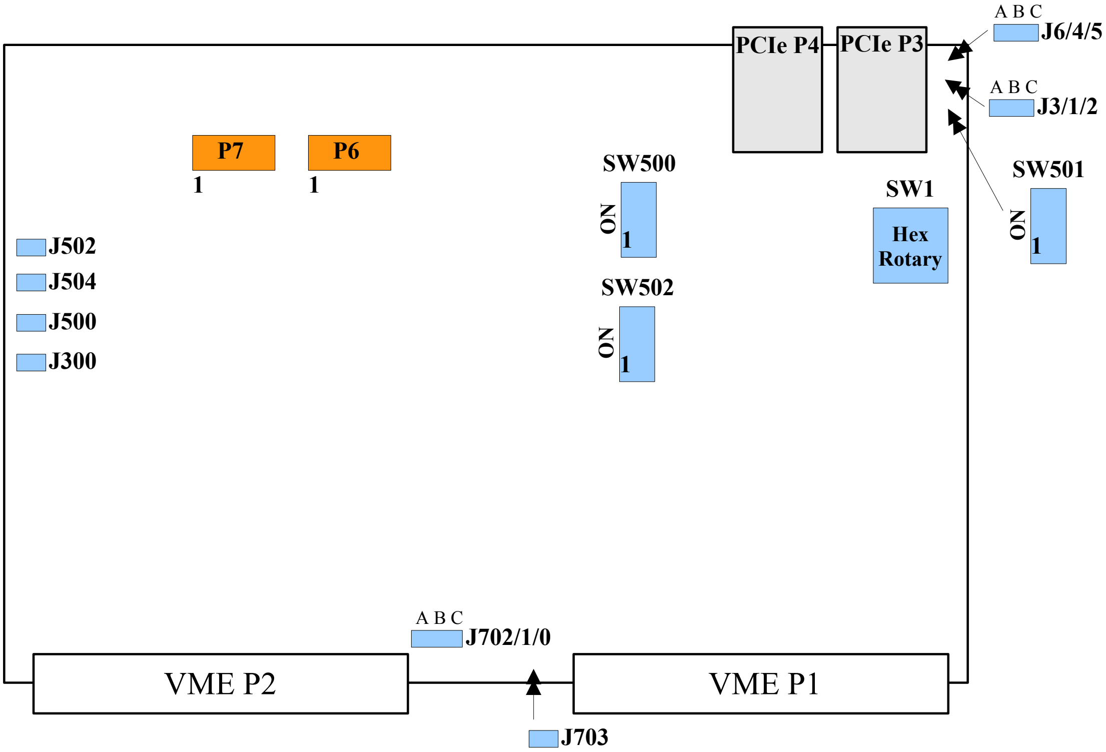
2 PEV_1100 Hardware Description

The following chapter provides a technical description of the hardware implementation. This covers following aspects :

- PEV_1100 physical implementation
- Hardware implementation of key sections
- FPGA internal architecture description

2.1 PEV_1100 Configuration

The above chapters provides the PEV_1100 static options and IO connectors description.



Drawing 2: PEV_1100 Component side

2.1.1 Static Options

Following tables sums up the PEV_1100 static options (Jumpers, Mini DIP switches and Hex-Rotary).

Jumper group	Valid position			

J[6-4-5]	J[4-6]	J[4-5]	J[6-4] = PCIe P4 DOWN_stream
			J[5-4] = PCIe P4 UP_stream
				PCI Express External.

J[1-2-3]	J[2-1]	J[2-3]	J[2-3] = PCIe P4 SBRTN
			J[2-1] = OPEN
				PCI Express External.

J[700-1-2]	J[700-1] (Top)	J[701-2] (Bottom)	J[700-1] = VME64V Power
			J[701-2] = On-board DCDC
				PMC 3V3 power supply selection

J703	Installed	Open	No on-board power	IOxOS Reserved
J300	Installed	Open	PMC JTAG by-pass. Used only when the PEV_1100 is under test.	IOxOS Reserved
J500	Installed	Open	Push Button RESET. Allow to control the on-board RESET.	IOxOS Reserved
J502	Installed	Open	JTAG 1149.1 Boundary chain Enable	IOxOS Reserved
J504	Installed	Open	JTAG 1149.1 P7 TAP OUT Enable. Used only when the PEV_1100 is under test.	IOxOS Reserved

Table 2.1.: PEV_1100 jumpers

Three mini-DIP switches implement the PEV_1100 configuration options :

- SW_500 implements the VME64X related options
- SW_501 implements the PCI Express External cabling options and the FPGA configuration options
- SW_502 implements the PCI Express mapping options and the Power-ON FSM sequencing control.

Switch	Default	Function	Comment
SW500-1	ON	VME SYSRST Enable	Enable VME SYSRESET generation
SW500-2	ON	VME SLOT1 Enable	Force VME Slot-1 active
SW500-3	OFF	VME64X Mode	VME64X CRCSR Mode
SW500-4	OFF	VME A24 A[19]	512 KB CRCSR window assigned statically while VME64X mode is disabled. In this case the CRCSR is mapped on the VME A24 Address Space. While VME64X mode is enabled and VME A24[23:19] = B"10101" the VME Auto_ID Mode is selected. While VME64X mode is enabled and VME A24[23:19] = B"10100" the VME Semi Automatic _ID Mode is selected. Refer to chapter 2.11.10
SW500-5	ON	VME A24 A[20]	
SW500-6	OFF	VME A24 A[21]	
SW500-7	ON	VME A24 A[22]	
SW500-8	OFF	VME A24 A[23]	
SW501-1	OFF	ON = SPI Prog Enable	Enable SPI Programming from P7 (Xilinx

Switch	Default	Function	Comment
			TAP Tool) Used for on-board FPGA bit-stream stored in the 128Mbit SPI FlashEPROM reprogramming
SW501-2	ON	FPGA Config B-STR LSB	Selection of Bit-stream #1 (Bit-stream #0 is the back-up)
SW501-3	OFF	FPGA Config B-STR MSB	
SW501-4	OFF	PCIe_RSTMOD	PCI Express remote RESET Mode = OFF : Disabled = ON : Enabled
SW501-5	OFF	P4 Cable Connector PCIe as Non-Transparent	When “ON” P4 is defined as UP_stream NT (Refer to 8624AA technical documentation)
SW501-6	OFF	This 3-bit field (SW501-8 is MSB and SW501-6) defines the PCI Express Infrastructure (OFF= '0', ON = '1') 000 : P3 UP_stream (external clock) 001 : P3 UP_stream (local clock) 010 : XMC#1 UP_stream 011 : XMC#2 UP_stream 100 : FPGA UP_stream 101 : P4 UP_stream (local Clock) 110 : Reserved 111 : PCIe Disabled	PEV_1100 on-board PCI Express implementation is defined through these three switched. Default Mode. (Required when SSC) Use local 100 MHz. (OK with CCG) PEV_1100 used as XMC carrier with installed CPU with PCI Express RC To be defined Back-up External cabling. Test Factory reserved for bring-up and testing.
SW501-7	OFF		
SW501-8	OFF = ”000”		
SW502-1	OFF	FPGA Configuration with Xilinx iMPACT tool.	Disable the CPLD FSM handling the PON FPGA configuration. When “ON” the FPGA shall be configured by the XILINX iMPACT tool.
SW502-2	ON	Enable Compressed 256 Bytes IO Space.	While enabled a 256 Bytes IO is used in place of the standard 4 KBytes
SW502-3	ON	NoPF_ BAR2_Size(0)	This 2-bit code define the A32 Non-Prefetchable Memory size associated with BAR_2. 00 64 MBytes 01 128 MBytes 10 256 MBytes 11 4 MBytes
SW502-4	ON	NoPF_ BAR2_Size(1)	
SW502-5	OFF	PF_ BAR0_Size(0)	

<table>
<tr><th>Switch</th><th>Default</th><th>Function</th><th>Comment</th></tr>
<tr><td>SW502-6</td><td>OFF</td><td>PF_ BAR0_Size(1)</td><td rowspan="2">This 3-bit code define the A32/A64 Prefetchable Memory size associated with BAR_0.
<table>
<tr><td>000</td><td>Disabled PF_ BAR0 and NoPF_ BAR2 selection.</td></tr>
<tr><td>001</td><td>Disabled PF_ BAR0 only.</td></tr>
<tr><td>010</td><td>128 MBytes</td></tr>
<tr><td>011</td><td>256 MBytes</td></tr>
<tr><td>100</td><td>512 MBytes</td></tr>
<tr><td>101</td><td>1024 MBytes</td></tr>
<tr><td>110</td><td>2048 MBytes</td></tr>
<tr><td>111</td><td>4096 MBytes</td></tr>
</table></td></tr>
<tr><td>SW502-7</td><td>ON</td><td>PF_ BAR0_Size(2)</td></tr>
<tr><td>SW502-8</td><td>OFF</td><td>PF_PCI_Express_A64</td><td>While “ON” Force the use of PCI Express 64_bit on BAR0 (A64 Capable)</td></tr>
</table>

Table 2.2.: Micro-switch PEV_1100 static options

2.1.2 Front panel LED Indicators

The PEV_1100 implements front panel LED indicators, providing visual information on main internal status.

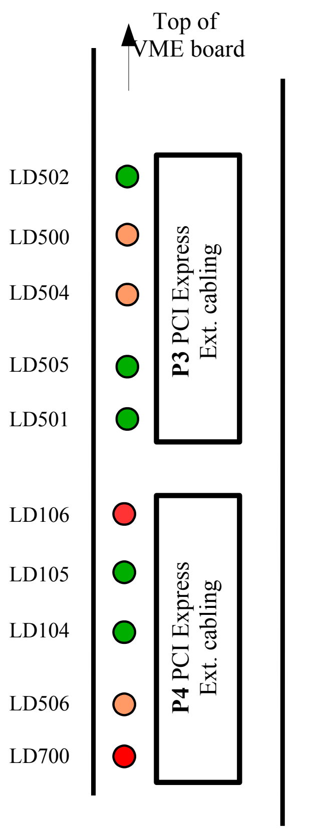
	LED	Col.	Meaning	
 Top of VME board LD502 LD500 LD504 LD505 LD501 LD106 LD105 LD104 LD506 LD700 P3 PCI Express Ext. cabling P4 PCI Express Ext. cabling	LD502	G	VME Slot_1 Enable	
	LD500	G	OK	
		R	ERROR. Internal activity ... Optional flashing FAST/SLOW	
		A	WARNING Internal activity ...	
	LD504	R	FPGA INIT Error. Light on when the FPGA configuration failed. Bad or corrupted bit-stream.	
		G	VME Activity. 0.3[s] Stretched light on VME Master activity. (Master/Slave)	
	LD505	G	... PCI Express configured, MSI issued Optional flashing FAST/SLOW	
	LD501	G	PCI Express Activity. 0.3[s] Stretched light while PCI Express ... Optional flashing FAST/SLOW	
	LD106	R	PEX8426AA Fatal Error	Refer to PLX 8624AA user's guide.
	LD105	G	PEX8426AA Port_PG0	
	LD104	G	PEX8426AA Port_PG1	
	LD506	G	Light_on : PEV_1100 Ready Fast flashing : Wait for IO&MEM Enable Slow flashing : Wait for 1st CONFIG	FPGA controlled LED
		R	Light_on : Wait for PLL lock-up Fast flashing : Wait for PCI Express LINKUP Slow flashing : TBD	
		A	Reserved debugging	
	LD700	R	TPS 2310 LI Controller / Circuit breaker. Light on while on-board CC detected.	

Table 2.3.: Front panel LED indicators

Note *Colour code is : G = Green, R = Red, A = Amber*

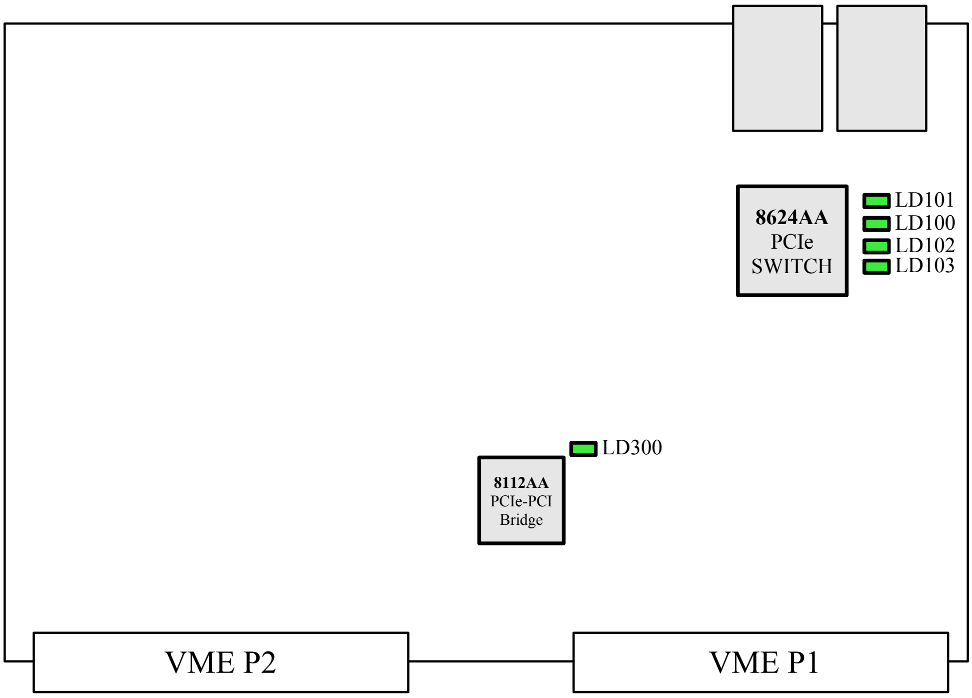
2.1.3 On-board LED Indicators

To support system integration, green SMD LED indicators are populated on the component side of the PEV_1100 PCB. Those provide PCI Express link-up status.

	Color	PLX Assignation	Meaning	Comments
LD101	G	PEX8624 Port_PG8	Provides PCI Express link status information on XMC Top	Refer to PEX8624AA user's manual for complete description. - OFF = PCI Express Link is down. - ON = PCI Express Link is up at 5GT/s Blinking schema refer to following section....
LD100	G	PEX8624 Port_PG9	Provides PCI Express link status information on XMC Bottom	
LD102	G	PEX8624 Port_PG6	Provides PCI Express link status information on PCI Express to PCI Bridge	
LD103	G	PEX8624 Port_PG5	Provides PCI Express link status information on Xilinx FPGA EP	
LD300	G	PEX8112	Indicates PCI Express "Link_UP". (… on solder side)	PEX8112 PCI Express to PCI Bridge is ready.

Table 2.4.: On-board (PCB) LED indicators

- Blinking 0.5s ON – 0.5s OFF, reduced Lanes are up at 5GT/s
- Blinking 1.5s ON – 0.5s OFF, all Lanes are up at 2.5 GT/s
- Blinking 0.5s ON – 1.5s OFF, reduced Lanes are up at 2.5GT/s



Drawing 3: PEV_1100 PCI Express LED indicators location

2.2 PEV_1100 Configuration process

Before installation, the PEV_1100 shall be configured (static option) in accordance with its field of utilisation.

The basic process is described in following steps :

(a) The PEV_1100 can be installed in legacy 3-row VME without backplane supplied 3.3[V]. While power hungry PMC/XMC are installed on the sites, the 3.3[V] power-supply of the XMC/PMC sites can be directly connected to the VME64X backplane power by configuring [J700-J701]. In other case leave the default setting [J701-J702]

(b) J703 is reserved for IOxOS debugging. While installed, the over-current logic is permanently activated. This jumper shall never be installed in normal operation..

(c) The PEV_1100 can be used in legacy VME64 (3-row connector) or in latest generation VME64X (5-row connector). The VME64X backplane provides extra signalling as geographical address and 3.3[V] power supply (…, not true in all implementation). The availability of geographic address allows to enable automatically the VME Slot_1 function (Central Arbitration and BTO). The mini-witch SW500-3 selected the mode of operation

- SW500-3 = “ON”, VME64X mode is selected.
- SW500-3 = “OFF”, VME64 mode is selected.

(d) In case of VME64 mode selected, the VME Slot_1 function shall be enabled/disabled with the specific mini-witch SW500-2.

- SW500-2 = “ON”, the on-board VME Slot_1 function is activated
- SW500-2 = “ON”, the on-board VME Slot_1 function is inactive. In this case the VME system needs an alternate unit acting as VME Slot_1 (? CPU). The PEV_1100 shall therefore be installed within the VME daisy-chain

(e) In case of VME64 mode selected, The 512KBytes VME Configuration Space, currently named CR/CSR is relocated to the standard A24 address space. The A24 base address is than determined with the mini-switch SW500-8 to SW500-4.

(f) The PEV_1100 is able to generate VME System RESET, during the power-up sequence and also under control of remote PCI Express command. The VME SYSRESET is enabled/masked with the mini-switch SW500-1

- SW500-1 = “ON”, PEV_1100 authorized to issue VME SYSRESET.
- SW500-1 = “OFF”, PEV_1100 not authorized to issue VME SYSRESET.

(g) The PEV_1100 is based on PCI Express External Cabling and implements two x4 connectors available on the VME front panel. These two connectors, P3 and P4 are configurable to act as

- PCI Express UP_stream port
- PCI Express DOWN_stream port

(h) P3 is a PCI Express UP_stream port and can only be dedicated to the Host connection.

(i) P4 can be configured to act as an DOWN_stream port for multi-chassis implementation (1st PEV_1100 P4 is controlling 2nd PEV_1100 P3 in a daisy chained manner) or also as a UP_stream port for a connection to a 2nd Host acting in Non Transparent Mode (NT)

(j) The SW1, 4-bit HEX rotary switch is directly attached to the front-end PCI Express switch device PLX PEX8624AA through its GPIO pins. This 4-bit static information is directly available to the software device driver for multi-chassis implementation. The HEX rotary ID is also available in the "ILOC_PCIE_SW" register

- SW1 Select a unique ID (from 0x1 to 0xF)

(k) The PEV_1100 implements a front-end PCI Express switch PLX PEX-8624AA. The assigned UP_stream port as well as the NT port are selectable with mini-switch SW501-5.

(l) Due to different PCI Express addressing capabilities, (AMD/INTEL, laptop/server/desktop) the default PCI Express BAR assignation can be modified through mini-switch SW502-8 to SW502-2. The PEV_1100 map two MEMORY windows (Prefetchable and Non-Prefetchable) with selectable size.

- SW502-4 and SW502-3 allows to select the size of the Non_Prefetchable BAR_2.
  - SW501-4 and SW501-3 = "OFF"-"OFF" : 64 MBytes.(Default)
  - SW501-4 and SW501-3 = "OFF"-"ON" : 128 MBytes.
  - SW501-4 and SW501-3 = "ON"-"OFF" : 256 MBytes.
  - SW501-4 and SW501-3 = "ON"-"ON" : 4 MBytes.
- SW502-7, SW502-6 and SW502-5 allows to select the size of the Prefetchable BAR_0.
  - SW501-7 to SW501-5 = "OFF"-"OFF"-"OFF" : Both disabled
  - SW501-7 to SW501-5 = "OFF"-"OFF"-"OFF" : Prefetchable BAR_0 disabled
  - SW501-7 to SW501-5 = "OFF"-"ON"-"ON" : 128 MBytes.(Default)
  - SW501-7 to SW501-5 = "OFF"-"ON"-"ON" : 256 MBytes.
  - SW501-7 to SW501-5 = "ON"-"OFF"-"OFF" : 512 MBytes.
  - SW501-7 to SW501-5 = "ON"-"OFF"-"ON" : 1024 MBytes.
  - SW501-7 to SW501-5 = "ON"-"ON"-"OFF" : 2048 MBytes.
  - SW501-7 to SW501-5 = "ON"-"ON"-"ON" : 4096 MBytes.
- SW502-8 select the 64-bit addressing on BAR_0. Not all PC chip-set support the 64-bit mapping.
  - SW501-8 = "OFF" : BAR0 is 32-bit capable.(Default)
  - SW501-8 = "ON" : BAR0 is 64-bit capable.

(m) The n-board SRAM based FPGA is configured at power-up with a specific bit-stream. The on-board SPI Flash memory stores up to 4 bit streams. Two mini-switch SW501-3 and SW501-2 allow to select which one is used for the power-up configuration. Specific tools are provided to load and manage the SPI Flash Memory with the FPGA bit-stream.

- SW501-3 and SW501-2 = “OFF”-”OFF” = Bit-stream #0.
- SW501-3 and SW501-2 = “OFF”-”ON” = Bit-stream #1.
- SW501-3 and SW501-2 = “ON”-”OFF” = Bit-stream #2.
- SW501-3 and SW501-2 = “ON”-”ON” = Bit-stream #3.

(n) Others static options, are reserved for services or debugging. In normal operation they shall be left to their inactive state “OFF”.

- SW501-1 = SPI Programming Enable. While “ON”, allows to use the XILINX iMPACT tool to re-program the SPI Flash Memory with a default bit-stream.
- SW502-1 = XILINX iMPACT . While “ON”, the FPGA configuration process is no more handled automatically but require external specific XILINX tool. Used during FPGA debugging.

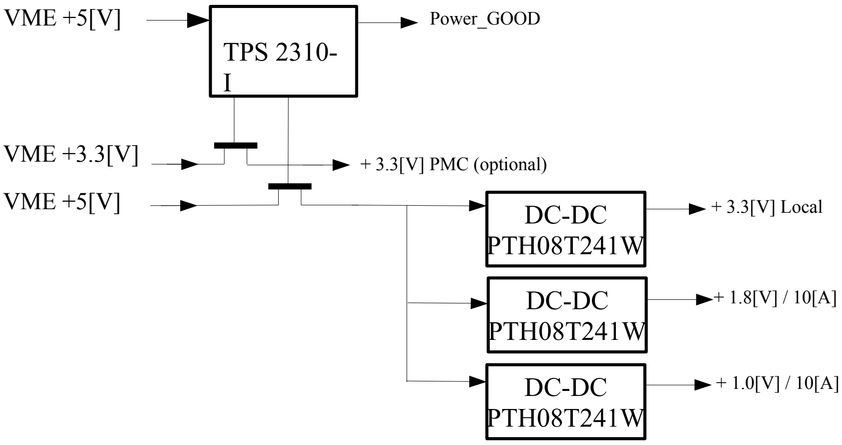
All mini-switch position status are available as read-only information in the register “ILOC_STATIC” register. Refer to chapter 3.2.1

2.3 Board Power Supply

The PEV_1100 board takes its power from the +5[V] VME backplane.

An electronic short-circuit breaker controller TPS 2310 protects the PEV_1100 against internal short-circuit, and monitors the power supplies. It controls two external DMOS for internal +5[V] and +3.3[V]. In case of power supply error detection (short circuit or power-up over-current), the front panel LED LD700 is light-on RED and the internal power are shut down.

The current limiter is calibrated to supply up to 10[A]on the internal +5[V] and 3.3[V] assigned to the PMC/XMC.



Drawing 4: PEV_1100 Power supplies Block Diagram

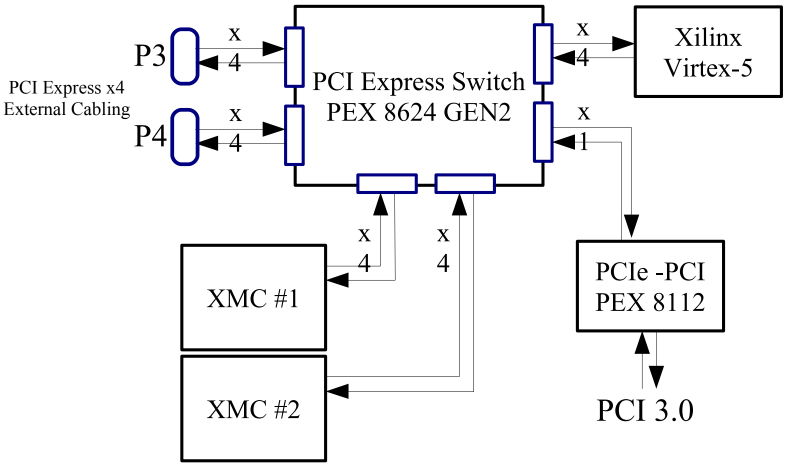
The on-board low voltage power-supplies 1.0[V], 1.8[V] and 3.3[V] are built with integrated DC-DC TI PT08T241W devices, able to supply up to 10[A] each. The DC-DC are dimensioned to support the complete family of XILINX Virtex-5 FF665 pins devices.

Others low-voltage/low-current power supplies are built with LDO LTC3026.

- Virtex-5 MGT Tx – Rx – PLL
- Local 2.5[V]

2.4 PCI Express Infrastructure

The PEV_1100 is based on a PCI Express infrastructure built around a PLX technologies PCI Express GEN2 Switch. This PCI Express centric is also deported to the FPGA internal architecture based on IOxOS Technologies TOSCA Network On Chip.



The on-board PEV_1100 PCI Express channels is distributed as :

- P3 & P4 PCI Express External Cabling
- XMC#1 & XMC#2 VITA 42.3 Mezzanine extensions
- PCI Express – PCI 3.0 Bridge
- FPGA Xilinx Virtex-5T FPGA

2.4.1 PLX Technologies PCI_Express Switch

The PLX PCI Express Switch requires some static assignations used for the system configuration. These assignation are made through the micro-switch SW501 (Refer to chapter 2.1.1)

- UP_stream Port : SW501- 8,7,6
- NT Enable : SW501 - 5

The UP_stream port can be assigned on any of the connected PCI Express device as long as it acts as a PCI Express RC. (Root Complex) This can be a Processor PMC or a XMC Mezzanine. In normal operation, The UP_stream Port is assigned to Port_0 (SW501- 8,7,6 = “000” or eventually “001”)

The PCI Express clocking schema requires specific care for correct operation. Two mode of operation are currently defined as :

- **SSC** “**S**pread **S**pectrum **C**locking”, introducing a +0 /- 5000 ppm / 30 KHz modulated
- **CFC** “**C**onstant **F**requency **C**locking ” introducing a +/- 100 ppm

For better EMI figure, the majority of PC chip set implements fixed **SSC mode**. (Some provide selection capability …) The on-board PCI Express Switch does not support mixed PCI Express clock as **SSC** and **CFC**. If **SSC** is used on the UP_stream, all PCI Express ports shall use the same clock reference and therefore the UP_Stream selection switch shall be set with SW501- [8,7,6] = “000”.

The PLX PCI Express Switch footprint can be populated with three different devices, providing thee, four or six PCI Express Port. Lower ports count devices could be installed in specific low-cost version.

PLX Device	PCIe Ports	Support	Comments
PEX8624XX	6	Complete implementation with PCI External Cabling P3 & P3, XMC 1 & 2, Legacy PCI Bridge and FPGA	Default implementation
PEX8616XX	4	PCI External Cabling P3 & P4, Legacy PCI Bridge and FPGA.	XMC VITA 42.3 ports removed
PEX8612XX	3	PCI External Cabling P3 & P4 and FPGA.	XMC & PMC ports removed

Table 2.5.: PCI Express Switch model

The PLX Technologies Switch version populated can be checked in the “ILOC_PCIE_SWITCH” register (Chapter 3.2.6)

Others PCI Express Switch status are also available in this register. This include the UP_stream port selection, the Non Transparent port selection and enable, and others related status.

To facilitate the PCI Express integration, LED indicator are provided to reflect the PCI Express link-up status on the six ports. The LED indicators status (light-ON, light-OFF or toggling) reflect the current PCI Express link

- Front panel LED LD105 and LD104 (Refer chapter 2.1.2)
- PCB internal LED LD100, LD101, LD102, LD103 (Refer chapter 2.1.3)

For complete technical information about the PCI Express Switch to “PLX Technologies PEX8624-AA_AB_BA Data book”

2.4.2 PCI Express External Cabling

The PEV_1100 is based on the “PCI External Cabling Specification” released by the PCISIG association. This standard is actually used in several high performance PCI Express applications. Cable in different length as well as legacy PCI Express card adapter are widely available.

The PEV_1100 implements two PCI Express x4 connectors (P3 & P4) located on the top of the VME board.

- **P3**, upper one default UP_stream port.
- **P4,** configurable as a regular DOWN_stream port, which can be used to cascade a 2nd PEV_1100 or as a Non-Transparent port, connected to a 2nd UP_stream port.

PCI Express External Cabling Connectors status are reflected in the “ILOC_Cable_1” and “ILOC_Cable_0” registers. (Refer to chapters 3.2.2 & 3.2.3)

2.4.3 PCI Express FPGA

The PEV_1100 implements a Xilinx Virtex-5T FPGA with embedded PCI Express EP. The FPGA is connected to the PLX PCI Express Switch Port #2 with a PCI Express x4 GEN1 interface.

The PCI Express EP status are reported in the "ILOC_PCIe_EP_STA" (Refer chapter 3.2.11) and "V5_PCIEP_RSLT" register. (Refer chapter 3.2.18)

The FPGA PCI Express link status is reflected by front panel LED LD506. Providing five (5) informations

LD506 Effect	Status informations
RED flashing fast (0.2[s])	Wait for link-up status from the PCI Express End point.
RED light-on	Wait for internal FPGA 100 MHz PLL is locked.
GREEN flashing slow (0.6[s])	Wait for 1^{t} PCI Express Configuration cycle. This is handled by the BIOS running the PCI enumeration.
GREEN flashing fast (0.2[s])	Wait for PCI Express IO and MEMORY Space enabled. This initialization can be handled by the BIOS or by the PEV_1100 device driver.
GREEN light-on	The PCI Express is fully initialized and ready to be used in all Spaces.

Table 2.6.: FPGA PCI Express link-up LED indicator

2.4.4 VITA41 XMC Extension

The PEV_1100 provides two XMC VITA42.3. Mezzanines. These two PCI Express x4 extensions are directly connected to the PCI Express Switch PEX8624 device.

Only XMC Jn15/Jn25 connectors , integrating the PCI Express ports , are implemented. For complete XMC specification refer to ANSI/VITA 42.3 – 2006.

XMC related status and control are provided in the "PMC_XMC_CSR" register (Refer to chapter 3.2.19)

2.4.5 IEEE 1386 PMC Extension

The PEV_1100 provides two PMC IEEE 1386 with PCI 32-bit 33/66 MHz.

The local PCI Bus is built with a PCI Express x1 to PCI bridge PLX PEX8112. This bridge device can be configured to a act as FORWARD or REVERSE mode. Others status and controls related to the PEX 8112 Bridge are located in the "PMC_XMC_CSR" register (Refer to chapter 3.2.19)

The PMC related user's IO connector Jn14/24 provides direct connection to/from

- Connection to FPGA Virtex-5 Bank_12 and Bank_8

- Direct connection to VME65X P2 (rows A and C)

Above table sums up the IO assignation.

PMC Signal allocation		Pin#		PMC Signal allocation	
Jn24 (Top)	Jn14 (Bottom)			Jn14 (Bottom)	Jn24 (Top)
fpga_GPIO_1	fpga_GPIO_33	1	2	fpga_GPIO_35	fpga_GPIO_3
fpga_GPIO_2	fpga_GPIO_34	3	4	fpga_GPIO_36	fpga_GPIO_4
fpga_GPIO_5	fpga_GPIO_37	5	6	fpga_GPIO_ 39	fpga_GPIO_7
fpga_GPIO_6	fpga_GPIO_38	7	8	fpga_GPIO_40	fpga_GPIO_8
fpga_GPIO_9	fpga_GPIO_41	9	10	fpga_GPIO_ 43	fpga_GPIO_11
fpga_GPIO_10	fpga_GPIO_42	11	12	fpga_GPIO_44	fpga_GPIO_12
fpga_GPIO_13	fpga_GPIO_45	13	14	fpga_GPIO_47	fpga_GPIO_15
fpga_GPIO_14	fpga_GPIO_46	15	16	fpga_GPIO_48	fpga_GPIO_16
fpga_GPIO_17	fpga_GPIO_49	17	18	fpga_GPIO_51	fpga_GPIO_19
fpga_GPIO_18	fpga_GPIO_50	19	20	fpga_GPIO_52	fpga_GPIO_20
fpga_GPIO_21	fpga_GPIO_53	21	22	fpga_GPIO_55	fpga_GPIO_23
fpga_GPIO_22	fpga_GPIO_54	23	24	fpga_GPIO_56	fpga_GPIO_24
fpga_GPIO_25	fpga_GPIO_57	25	26	fpga_GPIO_59	fpga_GPIO_27
fpga_GPIO_26	fpga_GPIO_58	27	28	fpga_GPIO_60	fpga_GPIO_28
fpga_GPIO_29	fpga_GPIO_61	29	30	fpga_GPIO_63	fpga_GPIO_31
fpga_GPIO_30	fpga_GPIO_62	31	32	fpga_GPIO_564	fpga_GPIO_32
VME_P2_A1	VME_P2_A17	33	34	VME_P2_C17	VME_P2_C1
VME_P2_A2	VME_P2_A18	35	36	VME_P2_C18	VME_P2_C 2
VME_P2_A3	VME_P2_A19	37	38	VME_P2_C19	VME_P2_C 3
VME_P2_A4	VME_P2_A20	39	40	VME_P2_C20	VME_P2_C 4
VME_P2_A5	VME_P2_A21	41	42	VME_P2_C21	VME_P2_C 5
VME_P2_A6	VME_P2_A22	43	44	VME_P2_C22	VME_P2_C 6
VME_P2_A7	VME_P2_A23	45	46	VME_P2_C23	VME_P2_C 7
VME_P2_A8	VME_P2_A24	47	48	VME_P2_C24	VME_P2_C 8
VME_P2_A9	VME_P2_A25	49	50	VME_P2_C25	VME_P2_C 9
VME_P2_A10	VME_P2_A26	51	52	VME_P2_C26	VME_P2_C 10
VME_P2_A11	VME_P2_A27	53	54	VME_P2_C27	VME_P2_C 11
VME_P2_A12	VME_P2_A28	55	56	VME_P2_C28	VME_P2_C 12
VME_P2_A13	VME_P2_A29	57	58	VME_P2_C29	VME_P2_C 13
VME_P2_A14	VME_P2_A30	59	60	VME_P2_C30	VME_P2_C 14
VME_P2_A15	VME_P2_A31	61	62	VME_P2_C31	VME_P2_C 15
VME_P2_A16	VME_P2_A32	63	64	VME_P2_C32	VME_P2_C 16

Table 2.7.: IEEE 1386.1 User's IO

2.5 FPGA Support & Infrastructure

The PEV_1100 main functionalities are implemented in a XILINX Virtex-5 FPGA. This **F**ield **P**rogrammable **G**ate **A**rray device provides a very efficient and versatile vehicle to support high performance digital logic. The complete PEV_1100 design is implemented in VHDL and fully simulated in adequate environment.

Thanks to the complete ownership of the embedded IP cores, and the re-programmability of the FPGA , IOxOS Technologies SA is able to upgrade or customize the PEV_1100 implementation when this becomes necessary.

Moreover some PEV_1100 FPGA sections are reserved for User specific implementation. A complete “FPGA Design Kit” is available to support this design effort.

- XILINX Virtex-5T FFG665 package.
  - LX30T, LX50T, SX30T, SX50T
  - FX30T, FX70T
- Local Configuration stream non-volatile memory storage. Refer to chapter 2.5.2
  - SPI Serial Flash 128 Mbit.
  - In-situ upgrade capability.
  - Up to 4 bit-stream stored in the SPI Serial Flash
- Complete FPGA design environment
  - VHDL source data base
  - Simulation environment with BFM

Each FPGA owns a 32-bit signature. This information used for FPGA tracing is implemented in hardware and built during the FPGA generation This information is available in “ILOC_SIGN” register. Refer to chapter 3.2.7

Because several FPGA implementations can be generated with different or selected functionalities, a 16-bit field is also provided in “ILOC_GENCTL” register. Refer to chapter 3.2.8

2.5.1 External CPLD

The PEV_1100 incorporates a non-volatile CPLD (Xilinx Coolrunner_II XC2C256), immediately operational after power-up. This device is also programmable in-situ with the Xilinx iMPACT tool. The CPLD handle following PEV_1100 functions :

- PEV_1100 board RESET management. (Voltage surveyor, watchdog, PCI Express external Cabling PERST, ...)
- FPGA Configuration with bit-stream stored in the Serial Flash EPROM
- Multiplexer scanning logic with the FPGA. This integrated function allows to reduce the required pin count in the FPGA device.

The FPGA Configuration FSM is fully integrated in the CPLD device. After a power-up RESET or a local Cold_RRESET, the FPGA is configured. The FSM read-out the defined bit-stream from the SPI Serial Flash device and push it in the FPGA device. The bit-stream number (#0 to #3) is determined with the static option SW501-[3:2].

The configuration bit-stream incorporates CRC and therefore incorporate error detection support. The FPGA Configuration FSM monitors continuously any error detected by the FPGA device. In case or error detected, the configuration process is restarted with bit-stream #0. (… which is the backup bit-stream)

A front panel LED indicator LD504 is light-on RED in case of FPGA configuration failure. This is a catastrophic error because the FPGA can not be configured. In this case, bit-stream #0 shall be loaded in the SPI Serial Flash trough its SPI programming port. Refer to chapter 2.6.3.

After configuration, status information are provided in the “ILOC_GENCTL” register. Refer to chapter 3.2.8 The selected bit-stream defined with SW501-[3:2] is also available in “ILOC_STATIC” register .

The FPGA configuration time is directly related to the size of the FPGA bit-stream. The process is handled at 50 Mbit/s. Following table sum-up the configuration time required.

FPGA Type	Configuration Size	Configuration Time
LX30T	9'371'136	~ 187 [ms]
LX50T	14'052'352	~ 281 [ms]
SX35T	13'349'120	~ 267 [ms]
SX50T	20'019'328	~ 400 [ms]
FX30T	13'517'120	~ 270 [ms]
FX70T	27'025'408	~ 540 [ms]

Table 2.8.: XILINX Virtex-5 FPGA Configuration size.

It is also possible to trigger remotely a new FPGA bit-stream reload through the “ILOC_SPI” register. Refer to chapter 3.2.5. During the reconfiguration, the FPGA becomes inactive and therefore the PCI Express connection is lost.

2.5.2 Serial Flash PROM

The PEV_1100 incorporates a 128 MBit Serial Flash EPROM (Numonix M25P128) used to store in non-volatile device the FPGA bit-stream and its associated PON_FSM micro-code. The device can be access from three different ports.

1. P7 programming port. Refer to chapter 2.6.3.
2. CPLD handling the power-up FPGA configuration.
3. SPI specific control register “ILOC_SPI” chapter 3.2.5

The SPI Serial Flash Device is fully accessible in run time and can than be reprogrammed from the Host controller with specific utilities. XprsMon command are provided to support new FPGA bit-stream programming (update).

The SPI Serial Flash is organized in two different mapping, fixed with option related to the bit-stream size of the on-board populated FPGA type. A status information “fpga_BIG” is provided in the register “iloc_GENCTL” Refer to chapter 3.2.8

For Virtex LX30T, LX50T, SX30T, FX30T :

FPGA bit-stream	Size	Offset area	Comments
Bit_Stream #0	16_MBit	0x00'0000 – 0x1F'FFFF	Default (Back-up) FPGA bit-stream
Bit_Stream #1	16_MBit	0x20'0000 – 0x3F'FFFF	Selected with SW501-[3:2] = “01”
Bit_Stream #2	16_MBit	0x40'0000 – 0x5F'FFFF	Selected with SW501-[3:2] = “10”
Bit_Stream #3	16_MBit	0x60'0000 – 0x7F'FFFF	Selected with SW501-[3:2] = “11”
Reserved	64 MBit	0x80'0000 – 0xFF'FFFF	Reserved for private storage

Table 2.9.: FPGA bit-stream mapping (Small devices)

For Virtex SX50T, FX70T :

FPGA bit-stream	Size	Offset area	Comments
Bit_Stream #0	32_MBit	0x00'0000 – 0x3F'FFFF	Default (Back-up) FPGA bit-stream
Bit_Stream #1	32_MBit	0x40'0000 – 0x7F'FFFF	Selected with SW501-[3:2] = “01”
Bit_Stream #2	32_MBit	0x80'0000 – 0xBF'FFFF	Selected with SW501-[3:2] = “10”
Bit_Stream #3	32_MBit	0xC0'0000 – 0xFF'FFFF	Selected with SW501-[3:2] = “11”

Table 2.10.: FPGA bit-stream mapping (Big devices)

The SPI non-volatile memory device can be programmed in-situ with the XILINX iMPACT tool. This programming procedure is used in case of all FPGA bit-streams are erroneous (… especially back-up bit-stream #0) Refer to chapter 2.6.2

2.5.3 FPGA System Monitor

The XILINX Virtex-5 FPGA incorporates a ADC section, allowing to implement monitoring on specific parameters. By default, internal parameters as VCC_{CORE} and die temperature are continuously evaluated. The XILINX System Monitor technical description is provided in UG192 document “Virtex-5 System Monitor User's guide”.

The PEV_1100 also provides external power supplies signals to the FPGA System Monitor.

System Monitor IN channel	Divide ratio	Monitored signals
IN_12	1:5.25	VME backplane +5[V]
IN_13	1:5.25	VME backplane +3.3[V]
IN_11	1:5.25	On-board DCDC +3.3[V]

Table 2.11.: FPGA System Monitor analog inputs

An external electronic reference, Burr-Brown REF3025, provides a precise 0.2% 1.0[V] voltage reference to the System Monitor.

The System Monitor is controlled trough “V5_SMON_ADDPT” , “V5_SMON_DAT” and “V5_SMON_STA”registers. Refer to chapters 3.2.14 to 3.2.16

2.6 Others General Functions

Following chapter provide technical description of miscellaneous functions.

2.6.1 I2C Controller.

The PEV_1100 integrates a I2C Controller allowing to interface directly following four devices :

- The PCI Express PLX PEX 8624 owns a I2C Slave port allowing to access internal registers. This I2C port can be used by the PON_FSM controller logic to initialize the PEX8624AA before the PCI Express RC starts its initialization process.
- The on-board temperature monitor LM86
- The two XMC IPMI SMB Bus (One per XMC VITA 46.3 slot)

The I2C Master controller is interfaced with four control registers “I2C_CTL”, “I2C_CMD”, “I2C_DATW” and “I2C_DATR” Refer to chapter 3.2.20

2.6.2 XILINX Programming Port P6

The PEV_1100 implements a 2[mm] 14-pin connector dedicated to the XILINX programming/ debugging tool Platform USB II. For full description of this tool refer to XILINX DS593 document. This connector is used for :

- FPGA Virtex-5 ChipScope PR
- FPGA Virtex-5 Configuration with Xilinx iMPACT tool
- CPLD a programming with Xilinx iMPACT tool

Signal allocation	TAP P6		Signal allocation
VREF	1	2	GND
SS/TMS	3	4	GND
SCK/TCK	5	6	GND
MISO/TDO	7	8	GND
MOSI/TDI	9	10	GND
NC1	11	12	GND
NC2 (TRSTn)	13	14	GND

Table 2.12.: XILINX programming Port P6

To use the P6 connector with the XILINX programming tool Platform USB II in the way to download the FPGA form the iMPACT utility, the mini-switch SW502-1 shall be “ON”.

While SW502-1 is “ON” the FPGA in not configured automatically at power-up with a bit-stream stored in the SPI Serial Flash but wait for a configuration bit-stream supplied through the P6 connection

2.6.3 SPI Serial Flash programming JTAG TAP J7

The PEV_1100 implements a second 2[mm] 14-pin connector dedicated to the JTAG 1149.1 and for the SPI Flash_EPROM programming Pins allocation is the same as P6.

The 1st (default) FPGA bit-stream, stored in the SPI Serial Flash_EPROM can be programmed with the XILINX iMPACT tool associated with the programming tool Platform USB II connected to P7. To support the SPI Flash_EPROM programming the SW501-1 shall be positioned “ON”.

While SW501-1 is “ON”, the SPI access to the Serial Flash_EPROM is reserved for the programming and therefore the FPGA Configuration with the CPLD is disabled.

2.6.4 Supervisor and Watchdog Timer

The PEV_1100 integrates a TPS3305, dual supervisor circuitry and watchdog timer. It provides following functions :

- 1.8[V] monitoring
- 3.3[V] monitoring
- Power-On RESET generation
- 1.6 [s] watchdog timer

The watchdog timer issues a new RESET every 1.6[s] in case of the FPGA is not configured.

2.7 TOSCA Central Switch

The TOSCA Central_SW (Central Switch) provides full mesh independent data flows. Each flow is scheduled independently from the others and therefore implements simultaneous transactions on all of them. By providing up to 8*1.6 = 12.8 GBytes/s, the 4-ports Central_SW is ready to support full speed PCI_Express x8, Revision 1.1 or PCI_Express x4, Revision 2.0.

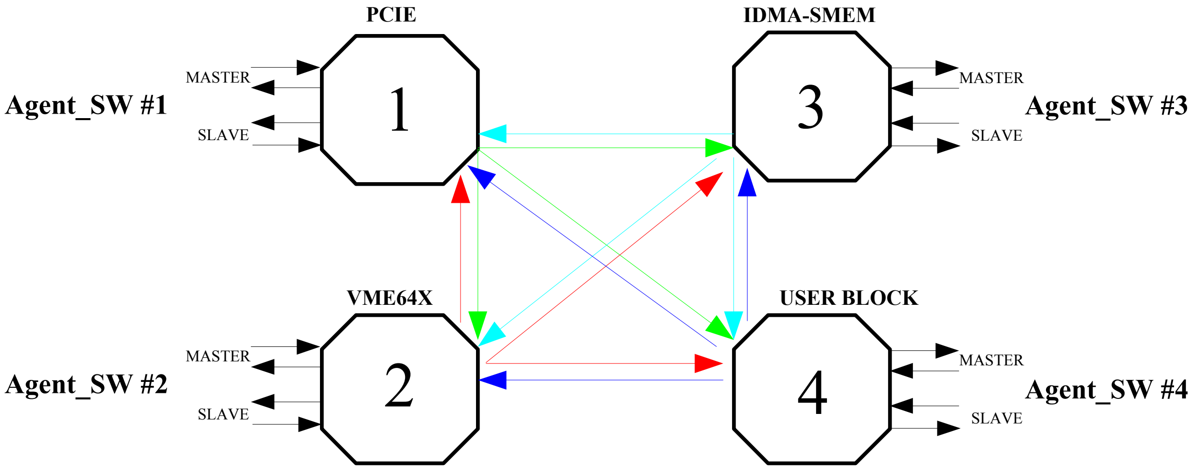
The two independent full mesh interconnection support following transaction type as :

- The WRITE Posting & READ_Request , 64-bit - 200 MHz, N independent path.
- The READ_Response, 64-bit - 200 MHz, N independent path.

The PV-1100 TOSCA Central_SW is composed of four independent back-end ports, each supporting Master/Slave + Read/Write independent channel access. By default, the TOSCA Central_SW supports data packets transactions up to 512 Bytes. The four ports are assigned as follow :

1. PCI Express, directly attached to the Virtex-5 PCI Express EP
2. VME64x
3. IDMAC – SMEM Shared Memory
4. USER Block

The Central_SW implements two full mesh interconnection between the four ports. Packet buffering is implemented at Source and at Destination sections, this independently for each type of transaction. The above diagram represents the four data flow for a transaction type (i.e WRITE Posting & Read_Request). A 2nd full mesh interconnect, (not represented) supports the READ_Response data flow...



Each Agent_SW port , integrates four interfaces types with specific buffering. The following table sums up the interface implementation.

Interface (.vhd)	Description	Buffering	Comments
WPOST_RDRQ_MAS	Write_Posting and Read_Request. The back-	6* [512+64] Bytes +	Two independent buffer queue segregating the Write_Posting and the Read_Request.

Interface (.vhd)	Description	Buffering	Comments
	end Master controller generates the packets	6* [64] Bytes	Specific control logic for the Write-Read ordering.
WPOST_RDRQ_SLV	Write_Posting and Read_Request. The back-end Slave controller consumes the packets and executes them.	3*4* [512+64] Bytes + 3*4* [64] Bytes	Independent buffer queue assigned for each sources
RDRS_MAS	Read Response with Data. The back-end Master controller, (initiator of the Read-Request) consumes the received data packet.	6* [512+64] Bytes	Provision made for "Write Status Response" in case of the Central-SW shall supports others protocol (i.e Rapid-IO or Specific)
RDRS_SLV	Read Response with Data.. The The back-end Slave controller load the response data packet.	6* [512+64] Bytes	Common buffering for the three Read Responder sources.

Table 2.13.: Agent_SW Transaction Interfaces

The PEV_1100 current implementation of the TOSCA Central_SW (4-ports, 512 Bytes data packet) consumes twenty (4* 5) 36K Block RAM. .

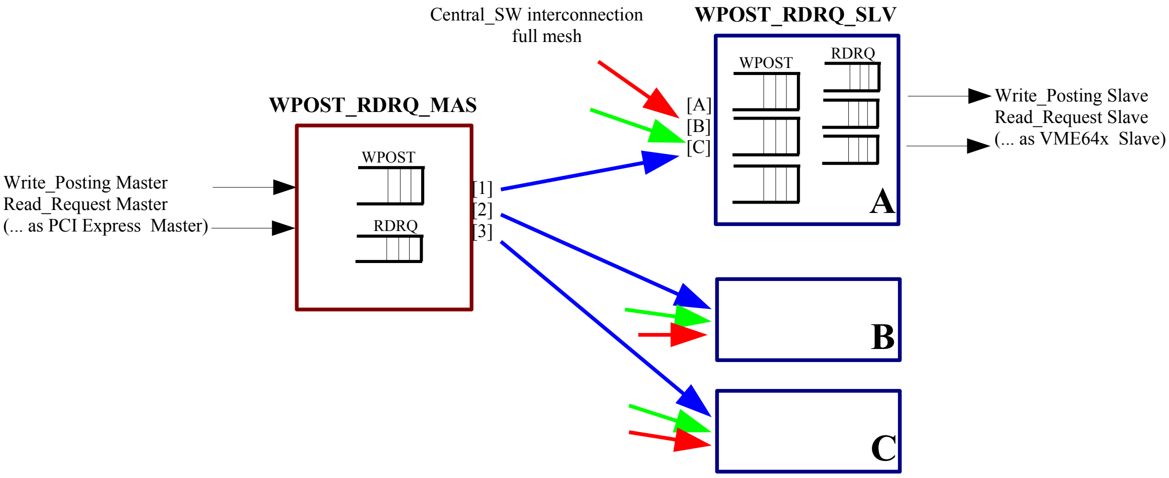
2.7.1 WRITE_Posting & Read_Request

525 The WRITE_Posting and the Read_Request transactions are sharing the same internal path and are supported through two independent interface blocks as :

- WPOST_RDRQ_MAS
- WPOST_RDRQ_SLV

These two blocks provides back-end interfaces and switching infrastructure connection. Both 530 interfaces are implemented in their own time-domain. Typically the switching infrastructure is running at 200 MHz and the back-end interface at its natural frequency.

The interconnections between the blocks are implemented with independent 64-bit, 200 MHz path. (full mesh schema)



Each WPOST_RDRQ_MAS block can move packet to three Destination WPOST_RDRQ_SLV blocks. TLP packets are moved at maximum speed (up to 1.6 GBytes/s) between two DPRAM buffer (Source DPRAM to Destination DPRAM).

Each WPOST_RDRQ_SLV controls the flow from the (N_Port -1) Source WPOST_RDRQ_MAS, by implementing an arbiter. This arbiter supports independent GRANT for the Read_Request and the Write_Posting.

2.7.2 Read_Response (RDRS)

The Read_Response transactions are implemented through an independent full mesh infrastructure. It is supported through two independent interface blocks as :

- RDRS_MAS
- RDRS_SLV

These two blocks provides back-end interfaces and switching infrastructure connections. Both interfaces are implemented in their own time-domain.

The interconnections between the blocks are implemented with independent 64-bit, 200 MHz path. (full mesh schema)

Each RDRS_SLV block can move packet to three Destination RDRS_MAS block. TLP packets are moved at maximum speed (up to 1.6 GBytes/s) between two DPRAM buffer (Source DPRAM to Destination DPRAM).

2.8 PCI Express TLP

The TOSCA infrastructure is fully derived from the PCI Express architecture. All TOSCA transaction packets implements the PCI Express TLP format. For information, following block diagram represents the PEV_1100 PCI Express EP interface.

		PCI Express TLI Rx		
Transaction Type	**PCIe Flow Control**	**BLK**	**Description**	**TC Policy**
Target IO_Read Request	NPH	B	IO transactions, 4 KBytes window bridged to the internal IO_Bus. Single cycle compelled, no pipeline support.	
Target IO_Write	NPH+NPD	B		
Target Memory_Write	PH + n*PD	E		
Target Memory_Read Request	PH	E	MEM transactions ,...	
Initiator Memory Read Completion	CPLH + n*CPLD	F		
Unsupported Transaction Target_CONF_Write/Read Target Message	Not used	A		
		PCI Express TLI Tx		
Initiator Memory Write	PH + n*PD	C		
Initiator Memory Read Request	NPH	C		
Target IO_Read Completion	CPLH +CPLD	B	IO transactions Completion. IO Read & Write require Completion. (3 or 4 DW)	

PCI Express TLI Rx				
Target IO_Write Completion	CPLH	B		
Target Memory_Read Completion	CPLH + n*CPLD	D		

Table 2.14.: TLI Transactions Dispatching

The XILINX Virtex-5(T) family integrates a HW IP_core, providing the primary layer of the PCI Express interface. This IP_core is interfaced through a bidirectional private TLI (Transaction Layer Interface)

- TLI Tx Interface, 64-bit 250 MHz.
- TLI Rx Interface, 64-bit 250 MHz.

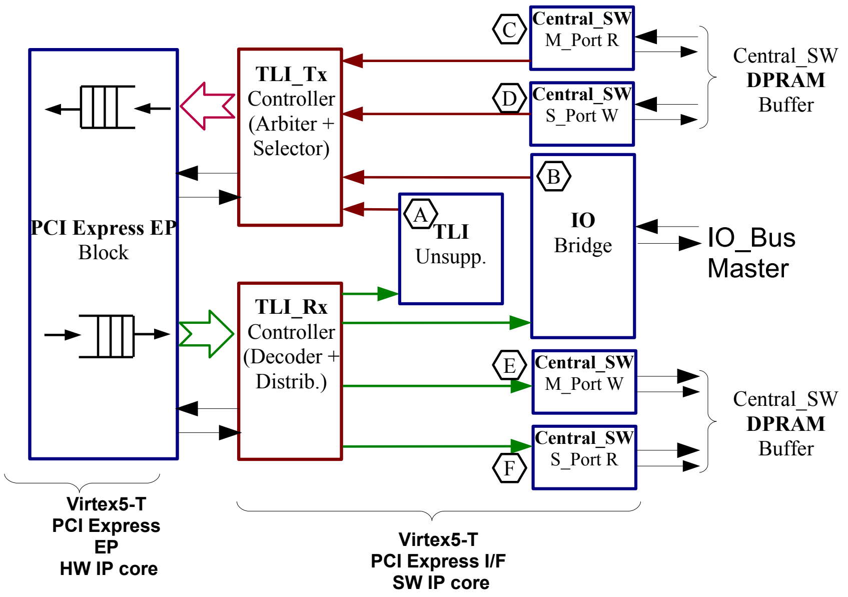
Each TLI Interface, (Tx – Rx) is handled with specific controllers assuring the dispatching, the arbitration, and the decoding of the in-going and out-going TLP. The two controllers are fully independent and able to process concurrently in-going and out-going TL

The Configuration **CfgRD0**, **CfgRD1**, TLP are decoded and handled by the PCI Express EP HW IP core, and therefore not handled by the local TLI controllers. Due to its “PCI Express End Point” characteristic, no Configuration TLP support is required.

A dedicated (mandatory) block handle the unsupported (or corrupted) in-going TLB. Proper error signalling is implemented in case of incoherent TLB.

IORd and **IOWr** TLP are handled by a dedicated block implementing a IO_Bus Master bridge function. The Completion **Cpl** or **CplD** TLP are issued for every IO TLP transaction.

The incoming (DOWN_stream) **MRd** and **MWr** TLP are routed to the TOSCA Central_SW through the Write_Posting and Read_Request buffers. Completions **Cpl** and **CplD** TLB are also routed back to the TLI Tx Interface.



2.9 SMEM Shared Memory

The DDR2 memory controller controls 4 devices, organized in x8. The PEV_1100 can provide 256, 512 or 1024 MBytes on-board DDR2 memory. The two bit field “ddr2_SIZE” located in the “SMEM_DDR2_CTL” register provides information on populated devices. Refer to chapter 3.4.1

ddr2_SIZE =	“01”	512MBit	= 256 MBytes (default)
	“10”	1024MBit	= 512 MBytes
	“11”	2048MBit	= 1024 MBytes

The DDR2 refresh strategy rate 3.9[us] or 7.8[us] and policy as single at the time or grouped in burst of eight are both selectable in the “SMEM_DDR2_CTL” register. Refer to chapter 3.4.1

The DDR2 memory controller implements four independent ports (labelled A, B, C, D) with segregated Read and Write access.

- Port_A : Central_SW Slave access.
- Port_B : IDMA Descriptor fetch and update.
- Port_C : IDMA data operation.
- Port_D : USER Block private access.

Each ports can be dynamically masked individually with ddr2_MASK[3:0]”control bits allowing to remove specific ports from the real-time operation. Refer to chapter 3.4.1

The four ports are arbitrated all together with selectable policy, allowing to tune for optimal application specific algorithm. For maximal performance, the Read and the Write are implemented through different channels. The arbitration algorithms are selectable in the “SMEM_DDR2_CTL” register. Refer to chapter 3.4.1

To support efficient READ, a dedicated read-ahead cache logic is implemented. When enabled, a full page (4 KBytes) is read-out from the DDR2 device and stored in a fast DPRAM. Two read-ahead 4 KBytes cache set/entry is implemented, selected with the LSM of the “Address Space Type”. This dual entry allows to sustain two full speed read-out process

A cache coherency protocol surveys and update the read-ahead buffer validity. Any WRITE to the tagged read-ahead address will invalidate the read-ahead buffer copy

The read-ahead caching mechanism support is conditioned by the “Address Space Type” field integrated in the Read_ Request transaction header. The “Address Space Type”is defined in the INgress MMU descriptors (PCI Express and VME64x)

Address Type	READ Mode of operation	Comments
0x0	Read-ahead cache disabled	Only the requested size is extracted from the DDR2 Memory
0x4	Read-ahead entry #0	A full 4 KBytes page is extracted from the DDR2 and copied in the read-ahead cache #0.
0x5	Read-ahead entry #1	Idem in read-ahead cache #0.
Others	Reserved	

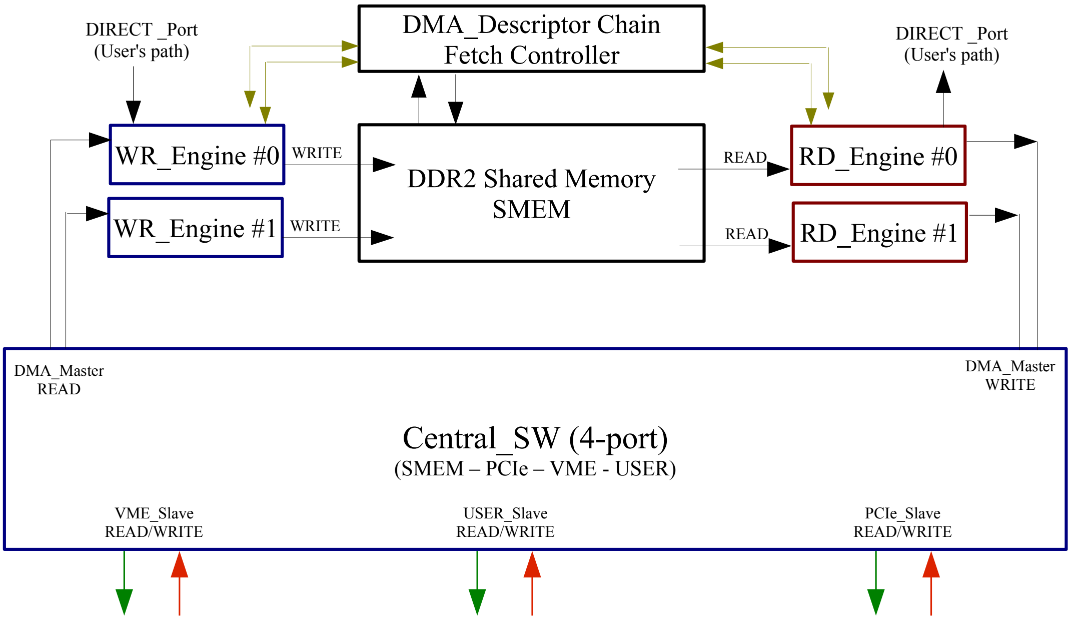
Table 2.15.: SMEM Central_SW TYPE transactions

Addressing error (out of range access) are monitored and error status flags are provided in the “SMEM_DDR2_ERR” register. Refer to chapter 3.4.2

2.10 IDMA Embedded Controller

The PEV_1100 implements a multi-channel (4) DMA controller, especially designed to support high efficient data movement between the Host PCI Express, the DDR2 shared memory and others local Bus interface.

The above drawing represents a simplified functional diagram of the IDMAC implementation. The IDMAC is directly attached to the SMEM block and therefore optimized to get maximum performance from the DDR2 devices. (burst size, page management, local DPRAM caches, ...)



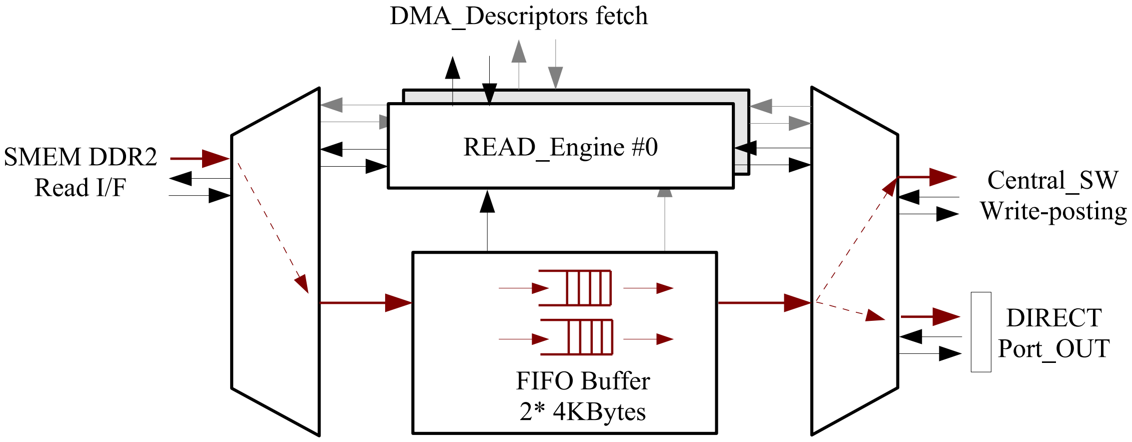
- The PEV_1100 IDMAC system, is implemented with independent READ and WRITE IDMA Engine. Each of the four engines can process its own chain of DMA_Descriptor.
- The DMA_Descriptor chaining shall be located in the local DDR2 SMEM. The DMA_Descriptor shall be first built by the application before any DMA process. To provide maximum performance, a cache memory (DPRAM) stores 64 consecutive DMA_Descriptor per channel.
- The IDMA SMEM DDR2 access (Read and Write) are optimized for block/burst transactions, supporting maximal bandwidth (up to 1.6 GBytes/s @ 200 MHz -DDR400)
- The IDMA WRITE_Engine(s) are dedicated for DMA transfer to the SMEM (This for a DDR2 write ...). Data source comes from any of the central_SW Agent (PCIe, VME64x or USER) or from the DIRECT Port_IN.
- The READ_Engine(s) are dedicated for DMA transfer from the SMEM (This for a DDR2 read ...). to any of the central_SW Agent (PCIe, VME64x or USER) or to the DIRECT Port_OUT.
- All IDMA Engines can be programmed with selectable “Start_of_Operation” trigger facilities as End of operation of another IDMA Engine, User's specific event, Global Time, ... Each DMA Engine can issue local Trigger on Start or End of a DMA_Descriptor

execution.

- Each IDMA Engine can issue INTERRUPT on the Start/End of a DMA_Descriptor execution. Reported Error can also issue specific interruption.
- DIRECT Port_IN/OUT are provided for on-chip DMA facilities to dedicated user's hardware as DAQ queues, autonomous communication engine, data servers, ...

2.10.1 READ_Engine Operation

The READ_Engine(s) are assigned to SMEM DDR2 Read operation, and therefore attached to the DDR2 controller through its dedicated READ channel. The two engines share the same physical interfaces to the SMEM, the Central_SW and the DIRECT Port_OUT. The 4 KBytes FIFO buffers are implemented in a single DPRAM block with simultaneous read/write access on the two ports.



Dedicated FSM manage concurrently the DMA Read transactions, (importing data to the FIFO_Buffer) and the Write transactions, (exporting data from the FIFO_Buffer). The Read_FSM masters the DMA execution (by filling the FIFO Buffer) while the Write_FSM is slave attached to the first one.

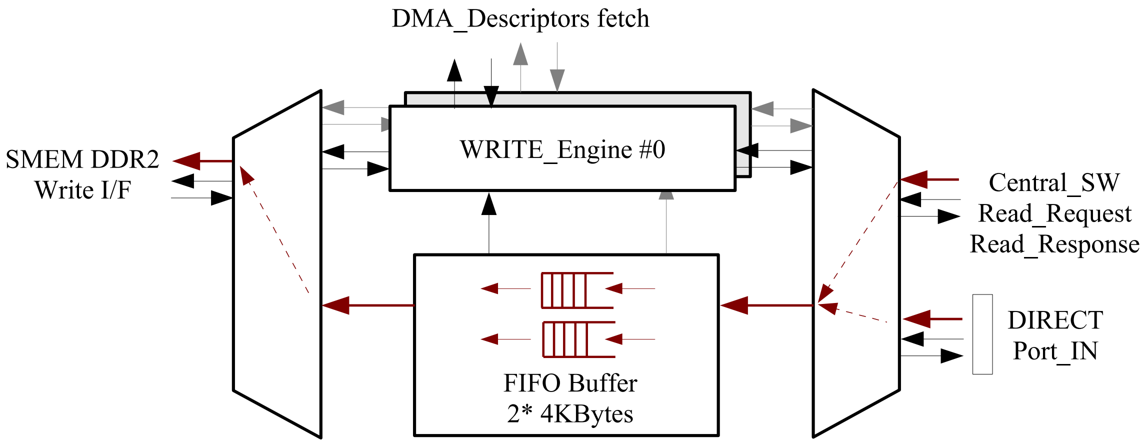
The 4 KBytes FIFO_Buffer has been dimensioned to provide optimal performance, by minimizing the dead-time on the interfaces. (Central_SW packets are 1 KBytes and DDR2 Page are 2 KBytes)

1. The SMEM DDR2 Read Interface (on the left side) is under control of the Read_FSM and mainly conditioned by space availability in the FIFO_Buffer. The DDR2 access transaction are fully predictable for data availability. Data read from the DDR2 memory are immediately transferred to the FIFO_Buffer at maximal transfer rate (1 64-bit QW per clock cycle)
2. The Write_FSM controls the Central_SW Write_Posting operation. Optimal transactions are applied when possible, with packet size up to 512 Bytes. PCI Express Target transaction are today limited by the INTEL/AMD Bridge to 128 or 256 Bytes maximal size.
3. The Write_FSM can also issue write to the DIRECT Port_OUT interface. This optionally provide a direct pathway to user's specific hardware.

Each channel can select independently its source and destination port (SMEM and Central_SW / DIRECT Port-OUT) for each DMA_Descriptor.

2.10.2 WRITE_Engine Operation

The WRITE_Engine(s) are assigned to SMEM DDR2 Write operation, and therefore attached to the DDR2 controller through its dedicated WRITE channel. The two engines share the same physical interfaces to the SMEM, the Central_SW, and the DIRECT Port_IN. The 4 KBytes FIFO buffers are implemented in a single DPRAM block with simultaneous read/write access on the two ports.



Dedicated FSM manage concurrently the DMA Read transactions, (importing data to the FIFO_Buffer) and the Write transactions, (exporting data from the FIFO_Buffer). The Read_FSM masters the DMA execution (by filling the FIFO Buffer) while the Write_FSM is slave attached to the first one.

The 4 KBytes FIFO_Buffer has been dimensioned to provide optimal performance, by minimizing the dead-time on the interfaces. (Central_SW packets are 1 KBytes and DDR2 Page are 2 KBytes)

The 4 KBytes FIFO_Buffer has been dimensioned to provide optimal performance, by minimizing the dead-time on the interfaces. (Central_SW packets are 1 KBytes and DDR2 Page are 2 KBytes)

1. The Central_SW Read Interface (on the right side) is under control of the Read_FSM and mainly conditioned by space availability in the FIFO_Buffer. The Central_SW Read is supported through Read_Request TLB and Read_Response. two dedicated FSM are handling the process completely decoupled. Data received with the Read_Response TLB are immediately transferred to the FIFO_Buffer at maximal transfer rate (1 64-bit QW per clock cycle)
2. Because the IDMA Central_SW Read are implemented with Read_request/Read-response, the performance is directly related to the number of outstanding Read_Request pipelined. Current implementation supports 1, 2, 3 or 4 outstanding Read_Request.
3. The Read_FSM can also controls the Read operation on the DIRECT Port_IN. In this case the split operation is disabled and replaced with a simple Bus based protocol.
4. The Write_FSM controls the SMEM Write operation. Optimal transactions are applied when possible, with packet size up to 1 KBytes.

Each channel can select independently its source and destination port (SMEM and Central_SW / DIRECT Port-OUT) for each DMA_Descriptor.

2.10.3 Some IDMA Special Capabilities

The PEV_1100 is optimized for autonomous data transport from and to the Shared memory SMEM DDR2. To provide extra facilities, two additional mode of operation have been implemented.

- DIRECT Mode
- PIPELINE Mode

2.10.3.1 DIRECT Mode

The DIRECT Mode provides a mapped bus interface between the IDMA engines and the USER Agent_SW. While enabled, (per channel) the operation usually assigned across the Central_SW are routed to a dedicated Interface of the user's FPGA area.. Only channel 0 supports DIRECT Port

Support of the IDMA infrastructure is simpler through the DIRECT Ports then across the complete Central_SW, requiring PCI_Express TLP implementation. Complete addressing informations allows to support multi FIFO or multi DPRAM with minimal FPGA resources and simple bus protocol. (Examples are provided in VHDL source code) Moreover, full bandwidth is supported thanks to direct interface with embedded FPGA SRAM.

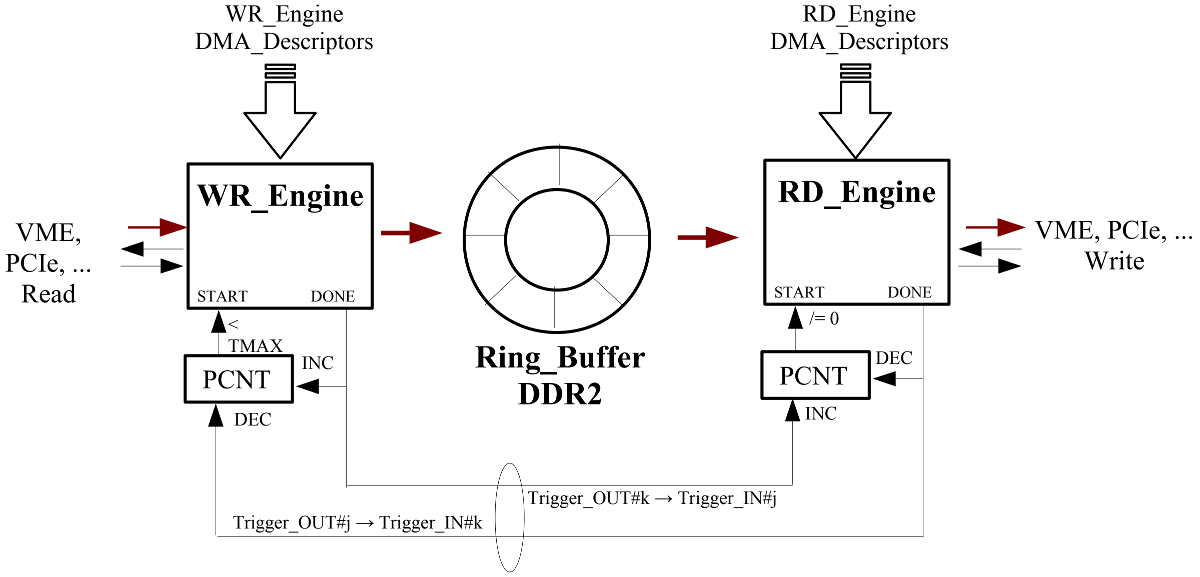
1. The DIRECT Port_OUT implements Write Bus protocol with complete Address + Data and associated synchronization. DIRECT Port_OUT is optimized for maximum throughput, no throttle is supported. Complete addressing information is supplied to the USER's Agent_SW block, including 64-bit Address and 16-bit Extended Address. This provides the end-user advanced addressing capabilities to support multi-channel queuing (FIFO) or buffering(DPRAM). The DIRECT Port_OUT is attached to the IDMA Read_Engine. (Reading data from the SMEM and send them to the DIRECT Port_OUT)
2. The The DIRECT Port_IN implements Read Bus protocol with complete Address + Data and associated synchronization. DIRECT Port_IN is optimized for maximum throughput, no data throttle is supported. Complete addressing information is supplied to the USER Agent_SW block, including 64-bit Address and 16-bit Extended Address. The DIRECT Port_IN is attached to the IDMA Write _Engine. (Reading data from the DIRECT Port_IN and send them to the SMEM)

The DIRECT Port_IN/OUT usage is supported through

- VHDL examples supplied with the TOSCA FPGA Design kit
- User's Block Agent_SW implementation with ad'hoc software example exercising the IDMA on the DIRECT Port_IN/OUT

2.10.3.2 PIPELINE Mode

The PIPELINE mode allows to cascade a WR_Engine with a RD_Engine with embedded ring buffer management facilities. The two engines use and manage a data buffer located in the DDR2 memory. Each engines integrates a 6-bit counter ("Trigger_PIPE_CNT" → PCNT) whose are used for the DMA_Descriptor execution control.



The WR_Engine reads the data source (VME64X, PCI_Express, others, ...) and store it to the DDR2 Memory. Start of its DMA Descriptors are enabled while the PCNT current value is smaller than 33. (ring buffer not full) At the end of the current DMA Descriptor execution, its PCNT is incremented and the corresponding RD_Engine PCNT is also incremented.

The RD_Engine reads the data source from the DDR2 Memory and store it to the destination (VME64X, PCI_Express, others, ...). Start of its DMA Descriptors are enabled while the PCNT current value is not equal to 0. (ring-buffer not empty) At the end of the current DMA Descriptor execution, its PCNT is decrementd and the corresponding WR_Engine PCNT is decremented.

This mode of operation provides seupport for user's transparent source to destination DMA operation. The DDR2 intermediate copy can be completely hidden. Thanks to the embedded facilities, DMA pipelined operation allows optimal performance on source and destination with minimal overhead time.

The PIPELINE mode is unabled and supported through dedicated registers (Refer to chapter 3.5.2 IDMA_XX_N_PCSR) and specific DMA Descriptor field **Start_Mode[2:0]**

2.10.4 IDMA Features Description

2.10.4.1 Independent READ/WRITE Engines.

The PEV_1100 implements 4 independent DMA controllers, as two READ engines and two WRITE engines. They are tightly coupled to the Shared Memory (DDR2 256 – 1024 MBytes) and therefore optimized to provide best performance matching with the DDR2 memories. The segregation between the READ and the WRITE engines provides several advantages as :

- The DDR2 interfaces are optimal for the on-going transaction type. The DDR2 Read out is optimized with local DPRAM used as Read-ahead Cache. The DDR2 Write is also supported through a front-end Write buffer.
- The DMA READ Engine Read-ahead Cache is supported with a 4 KBytes DPRAM allocated as 2* 2 KBytes per engine.
- The DMA WRITE Engine write buffer is supported with a KBytes DPRAM allocated as 2* 2 KBytes per engine.
- The same DMA_Descriptor chains can be executed by READ or WRITE engine, with full symmetry.
- Synchronization mechanism is provided, allowing to trigger (start) a WRITE Engine on end of execution of a WRITE Engine. (...or any others type of synchronization)

DMA controller supporting complete Source/Destination capabilities need to implement intermediate data buffer, with mismatch performance on the READ and WRITE section. By implementation is is always difficult to match the specific requirements on the READ and on the WRITE side at the same time.

2.10.4.2 DMA Address Control .

The PEV_1100 DMA controller provides optional DMA Address management. As for any DMA processor, the DMA Engines provide a source address reference (READ side) and a destination address reference (WRITE side). usually the source and the destination address are incremented independently related to the data operand handled. (Byte, half-word ...)

The PEV_1100 DMA Engines provide additional Address management support, allowing to support optimal data collection.

- The local Address (Shared memory) can be optionally not updated with the value of a new DMA Descriptor. The previous current value can be kept, allowing to support contiguous data across several DMA Descriptors.
- The local Address (Shared memory) can be optionally not updated with the value of a new DMA Descriptor but added with. This allow to built DMA_Descriptor chain with address relative.
- The external Address (Central_SW) can also be configured for no-update.
- The external Address (Central_SW) can be kept as fixed., useful for interfacing with FIFO based data interface.

2.10.4.3 PCI Express support.

The PEV_1100 DMA controller are designed to access the PCI Express infrastructure. PCI Express operation require specific conditioning.

- The PCI_Express TLP optional control bits SNOOP, RELAX and AT, can be defined in the DMA_Descriptor DW_0. By default keep these option deserted. Refer to PCI Express specification before setting one of them.
- The PCI Express TLP size are conditioned by system parameters, usually defined by the Host BIOS. The two following parameters are used by the IDMA addresing the PCI_Express Agent_SW.
    - MAX_PAYLOAD_SIZE determining the maximal length of any TLB with data. (Write_Request and Read_Completion with data)
    - MAX_READ_REQUEST_SIZE
- The number of maximal outstanding Read_Request can be defined in the DMA_Descriptor DW_0. This 2-bit field allows to set-up for 1,2,3 or 4 outstanding Read_Request. Maximum performance will be achieved with 4 outstanding Read_Request.

2.10.4.4 Central_SW Addressing .

The PEV_1100 DMA_Descriptors DW_1 implements the Central_SW routing with 4-bit field. Only the two LSB are actually used.

00**00** = PCI_Express Agent_SW

00**01** = VME64X Agent_SW

00**10** = SMEM-IDMA Agent_SW (Not usable for IDMA operation)

00**11** = USER Agent_SW

2.10.4.5 Global Time Stamping.

The PEV_1100 DMA implements a dedicated mechanism allowing to tag (time-tagging) the IDMA DMA_Descriptor operation. At the end of execution of a DMA_descriptor, the executed DMA_Descriptor DW_6 and DW_7 are updated with the Global_Time current value. (Write over the current DMA_Descriptor in the SMEM) This optional mode of operation allows to tag the IDMA operation for performance monitoring or for execution profiling.

The PEV_1100 Global time is located in the the VME64X Agent_SW block, Refer to chapter 2.11.7

2.10.4.6 Trigger facilities.

The PEV_1100 DMA Controller implements external trigger capability. While a DMA_Descriptor is fetched in the corresponding Engine, its execution can be delayed until a pre-defined trigger condition happens. This Start on Trigger condition is defined in the DMA_descriptor DW_3.

000 = Immediate execution. (No external trigger condition expected)

001 = Start execution at Global_Time Synchronization.

010 = Start execution at GPIO Event#1 (USER Central_SW block)

011 = Start execution at GPIO Event#2

100 = Start execution at IDMA on Trigger #0 event (READ_Engine #0) or if Pipeline mode is enabled in the PCSR Register (Trig_PIPE_Enable = '1')

- Start execution RD_Engine while Trigger_PIPE_CNT < “100000”
- Start execution WR_Engine while Trigger_PIPE_CNT /= “000000”

Start execution at IDMA while Trigger_Pipe_CNT /= X”00”.

101 = Start execution at IDMA Trigger #1 (READ_Engine #1)

110 = Start execution at IDMA Trigger #2 (WRITE_Engine #0)

111 = Start execution at IDMA Trigger #3 (WRITE_Engine #1)

Every IDMA Engine can also issue a Trigger event, selectable in the DMA_Descriptor DW_0 :

- DMA_Descriptor End of execution
- DMA_Descriptor Start of execution

These four trigger conditions can be used as Start Trigger condition for any pending DMA_Descriptor, allowing to built IDMA chaining with synchronization mechanism between the IDMA Engine.

i.e : IDMA READ_Engine #0 can executes a multiple DMA_Descriptor chain [Data collection from several VME64X module] and at the end ,trigger the start of WRITE_Engine #0 [Copy of the acquired data to the Host System Memory across PCI_Express]

2.10.4.7 DMA_Descriptor Location.

The PEV_1100 DMA Controller requires that the DMA_Descriptor chain shall be located in the SMEM Memory. Chain of DMA_Descriptor can not be located across the Central_SW interconnect. (i.e residing in the Host System Memory. This way of doing has been selected to have a deterministic access to the DMA_descriptor storage resources.

The IDMA controller is built to own a private access path to the SMEM DDR2 Memory. To improve further DMA_Descriptor fetch latency, a DPRAM Cache is implemented for each of the IDMA Engine. These DPRAM cache (4 entries) allow to reduce the dead-time of the DMA_Descriptor fetch sequence under 50 [ns].

- 2* 1KBytes (32 DMA_Descriptor) for WRITE_Engine channels
- 2* 1KBytes (32 DMA_Descriptor) for READ_Engine channels

If end user application requires DMA_Descriptor chain built in the Host System Memory, a READ_Engine can be set-up to copy the DMA_Descriptor chain located in the Host System Memory to the PEV_1100 SMEM and at the end of the copy automatically start the execution of the concerned IDMA chain. Several scenario can be set-up in advance, and ready to be executed on demand.

2.10.4.8 IDMA Error Signalling.

The IDMA engines issue Read and Write transaction on three attached slave devices. the above description sums up the potential error condition.

The SMEM DDR2 occupies a fixed size, (256, 412 or 1012 MBytes) over a 32-bit (4 GBytes) addressing. IDMA transactions outside the decoded area are immediately terminated with an error condition. This error is detected while the DMA_Descriptor is analysed before its execution. IDMA operation terminated with this error condition is signalled with status information “idma_STATUS[2:0] = 110”

The IDMA Write_Posting transactions to the Central_SW does not support any error signalling. Due to its nature, the transaction initiator does not get any information from the target about error condition. Error status informations are usually provided by the target interface (i.e PCI Express EP or VME64X Master)

The IDMA WRITE_Engine Read_Request / Read_Response transactions issued to the Central_SW implements several status error. To its split transaction implementation, any Read_Request requires specific Read_Completion (s). Any abnormal operation will be detected and signalled appropriately in the IDMA_GCSR register. The current DMA_Descriptor is aborted immediately on error detection.

- Unexpected Read_Response TLB received.
- Malformed Read_Response TLB received.
- Out of sequence Read_Response TLB received.

Moreover, signalled error (with Completion Status = b”100”) are also handled appropriately. In case of the control-bit option “**ext_DMAERR**” is set, the DMA chaining is not stopped, current DMA_Descriptor is aborted and next DMA_Descriptor execution is started. With the DMA_Descriptor Update capability, the abort DMA_Descriptor word counter can be saved.

2.11 VME64X Interface

The PEV_1100 provides a complete VME64X Master/Slave Interface with Slot_1 function. The following sections provide an overview of the implemented VME sub-functions. For complete VME Bus understanding please refer to the VITA specification with their related addendum.

The PEV_1100 is ready to operate in VME64X 5-rows backplane as well in legacy VME64 3-row backplane. Static option SW500-3 determines the mode of operation.

The PEV_1100 VME Interface is ready to support VXI related protocol.

2.11.1 VME Slot_1 functions

Any VME system requires a centralized function, commonly named Slot_1 and usually located on the left most slot of the backplane. While the VME Slot_1 in enabled, the LED indicator LD502 is light on GEEEN.

The VME64X geographic address, as well as others related static options are available in the "PVME_SLOT" register .

The VME Slot-1 function is enabled manually with static option while the VME64X mode is disabled. Refer to chapter 2.1.1 for static option selection or while the PEV_1100 is plugged in VME Slot No "1" defined by the VME64X 5-rows backplane GA+GAP Geographic addressing.

While enabled the PEV_1100 provides a VME Central Arbiter. Mode of operation as PRI, RRS are selectable in the "PVME_SLOT" register. A 4 [us] arbitration time-out is specifically assigned to the BGOUT assertion.

While enabled the PEV_1100 provides the BTO / 2eBTO sub-function. Any VME on-going cycles are monitored and in case of non-responding after a specified time generates a VME BERR#. The time-out value (from 16 to 128 [us]) and the associated flags can also be selected in the "PVME_SLOT1" register. Refer to chapter 3.3.1

The VME SYSRESET# signal can be controlled by the on-board PEV_1100 logic. A specific static option SW500-1 enables this capability. The VME SYSRESET# generator is automatically activated at power-up or can also be issued remotely through the "ILOC_SPI" register. Refer to chapter 3.2.5

2.11.2 VME Slave Port

The PEV_1100 can be set-up to act as a VME Slave agent. Two selectable window can be mapped respectively in the CONFIG/A24 and A32 address space. The VME A64 mode is not supported.

- CR/CSR or A24 512 MBytes window.
- Programmable A32 16 to 2048 MBytes window.

A 512 KBytes window, following the VME64X CR/CSR specification is mapped over the CR/CSR Space (AM = 0x2F) while the PEV_1100 is set-up to operate in 64X Mode with SW500-2. When the 64X mode is disabled, the same CR/CSR window is mapped on the VME Standard A24 Address Space A24. The base offset is then defined by the static option SW500[8:4] in place of the geographical addressing.

The 512 KBytes CR/CSR Space (Configuration ROM and Control & Status Register) is fully defined by the ANSI VITA 1.1 "VME64 Extensions" The CR/CSR 512KBytes window implements as defined by the VITA specification. CSR_BAR, CSR_BSR/BCR and CSR_ADER registers are described in chapter 3.3.13, 3.3.14, 3.3.15.

The power-up value of these concerned registers can also be set-up by the PON_FSM unit. Refer to chapter

In addition to the VME64X related registers, a IO_Bus Bridge unit allows to remap transparently all PEV_1100 IO_Bus mapped resources on the VME Bus.

The 2nd window is mapped on the VME Extended address Space A32. Its size is programmable from 16 MBytes up to 2 GBytes. This window is divided in pages and associated to an VME INgress MMU. Each page is associated to a Page Descriptor providing missing addressing information (Agent_SW and 64-bit Address extension) an some additional selectable options.

The A32 Window size is defined in the “PVME_MASCSR” register (selectable from 16 to 2048 MBytes) as well as other key operational option. The VME Slave Interface can also be globally disabled/enable. Following the VME64X specification, the base offset of the A32 window is defined by the CSR_ADER and enabled in the CSR_BSR/BCR register. Refer to chapter 3.3.3

The PEV_1100 integrates a MMU type address remapping function, named INgress MMU. The A32 decoded window is divided in N* pages of 1/2/4 MBytes and each page is directly associated to a 64-bit Page_Descriptor described in chapter 3.3.6. The INgress MMU is implemented with a local DPRAM storing the complete entries. This DPRAM look-up table can be initialized by the Host controller through specific registers “PVME_MMUDAT” and “PVME_MMUADD”**.** Through the INgress MMU, protection mechanism as well as complete remapping as well as hardware based swapping is supported .

The PEV_1100 slave interface supports all VME data transfer SLT, BLT, MBLT, 2eVME and 2eSST(all rates) across the VME A32 Addressing space. Access to the CR/CSR is limited to the SLT.

The PEV_1100 also integrates 4 Location Monitor registers. These resources can be mapped separately anywhere in the VME A16, A24 or A32 through “PVME_LOCMON” registers. Access over them can be programmed to issue interrupt to the Host controller . Refer to chapter 3.3.9

The Local VME64X Interrupt Generator controlled by the PVME_INTG register can be used to issue local VME Interrupt. Refer to chapter .

2.11.3 VME Master Port

The VME64X integrates a full VME Master Interface implemented as a bridge function. The local Central_SW transactions are converted to VME Bus cycle(s). Any of the on-chip Agent_SW (PCI Express EP, IDMA, USER) can than drive the VME Master Interface.

The VME Master interface is controlled through the “PVME_MASCSR” register. Refer to chapter 3.3.2.   The VME arbitration level, the arbitration release mode and other directly related options are made available to the user's application through the “PVME_MASCSR” register.

The VME Master is acting as a bridge function, therefore the complete cycle parametrization is built by the transaction source agent. The VME addressing mode id defined by the TYPE 4-bit field which is generated by the INgress MMU (for PCI Express EP) or directly by the IDMA Descriptor.

The User/Supervisor can be optionally selected in the “PVME_MASCSR” register. Following table sums-up the VME TYPE[3:0] remapping

Space_Type	VME64x Master	VME AM code
0x0	Configuration Space	AM = 0x2D
0x1	A16 : D08-D16-D32	AM = 0x29 / 0x2D
0x2	A24 : D08-D16-D32	AM = 0x39 / 0x3D
0x3	A32: D08-D16-D32	AM = 0x09 / 0x0D
0x4	A32: BLT	AM = 0x0B / 0x0F
0x5	A32: MBLT	AM = 0x08 / 0x0C
0x6	A32: 2eVME	AM = 0x20 XAM = 0x01
0x7	A32: 2eVME Fast	AM = 0x20 XAM = 0x01
0x8	A32: 2eSST_160	AM = 0x20 XAM = 0x11
0x9	A32: 2eSST_233	AM = 0x20 XAM = 0x11
0xa	A32: 2eSST_320	AM = 0x20 XAM = 0x11
0xb	A32: 2eSST_400	AM = 0x20 XAM = 0x11
0xc	Reserved	
0xd	A24: Address Only	AM = 0x39 / 0x3D (ADOH AM = 0x35)
0xe	A32: Address Only	AM = 0x09 / 0x0D (ADOH AM = 0x05)
0xf	IACK Cycle	AM = 0x3F

Table 2.16.: VME Master Central_SW TYPE transactions

The PEV_1100 supports all VME data transfer SLT, BLT, MBLT, 2eVME and 2eSST(all rates) across the VME A32 Addressing space. Optimal data transfer mode is selected in real time by hardware logic.

2.11.4 VME Location Monitor

The PEV_1100 implements 4 Location Monitors. A base register PVME_LOCMON holds a 32-bit address field A[31:5] and two control bit for the mode of operation. When a VME cycle match the specified Address (A[31:5] for A32, A[23:5] for A24 or A[15:5] for A16) a local interrupt is issued , selected by Address[4:3] field.

The four Location_Monitor interrupts are wired to the local ITC16 VME Interrupt controller. The PEV_1100 can issue VME cycles for location monitors with ADO or self addressed regular VME SLT cycles.

2.11.5 VME LOCK Support

The PEV_1100 supports VME64X ADOH transactions used to implement the VME LOCK. This specific mode of operation is dedicated to support VME atomic sequence with VME Slave reservation. (Refer to VME64X specification. This mechanism is based on following sequence :

1. An Address Only with Handshake ADOH cycle, issued by a VME Master, triggering the LOCK process. This transaction is normally issued by a native Read transaction. (CPU reads from ADOH location)
2. Access to the VME Master is first controlled by an arbitration phase. The triggered ADOH transaction is issued on the VME Bus only while the mastership arbitration is granted.

3. The VME Slave decoding the ADOH shall respond to the VME transaction. Three types of handshake can be generated as :
    - DTACK#, signalling acceptance of the LOCK process. While happens, the VME mastership is kept (BBSY# remains asserted) until a specific action (described in following section) will force the VME mastership to be released.
    - BERR#, signalling an error condition. Impact on the LOCK process is left open by the VME specification. A control bit “vmas_BACKOFF” located in the PVME_MASCSR Register allows to force PCI Express Completion without error.
    - RETRY#, signalling failure of the LOCK process. While happens, the VME mastership is released and the ADOH transaction is reissued. A control bit “vmas_MAXRETRY” located in the PVME_MASCSR Register allows to limit the number of retries.
4. While LOCK is won, the VME mastership is kept by the VME Master who granted the LOCK. This is supported by keeping BBSY# asserted. As long as the LOCK is active, the VME Master can issue any number of Read / Write transactions to the VME Bus. In practice the LOCK mechanism is used to emulate atomic transactions as Read Modify Write or Compare and Swap.
    - In case of BERR# handshake detected during the LOCK process activated, BBSY# is deactivated and therefore LOCK is immediately deactivated.
5. The LOCK is deactivated with two selectable mode of operation.
    - Set the control bit “LOCK_CLEAR_Cmd” located in the PMAS_LOCK Register. (PCI Express IO Space)

Executes a ADOH (or ADO) with a Write operation. This transaction will not be executed on the VME backplane, but will be used to clear the LOCK flag.

2.11.6 VME Error Support

The PEV_1100 integrates an error monitoring logic. A couple of Error registers “PVME_STAERR” and “PVME_ADDERR” provide basic VME debugging support.

In case of VME Error detected, while the PEV_1100 is acting as VME master the main VME cycle informations are stored in these two registers . These informations can be read-out by the Host controller for further analysis. Refer to chapter 3.3.7 and 3.3.8

2.11.7 VME Interrupts

The PEV_1100 fully integrate the VME Interrupt infrastructure as Interrupt generator and Interrupt Handler.

The Local VME64X Interrupt Generator controlled by the “PVME_INTG” register . Refer to chapter 3.3.4

The PEV_1100 integrates a complete VME INTH function. To support efficient Interrupt handling, autonomous IACK generation is supported by hardware. The VME INTH is associated to the VME dedicated ITC16 Interrupt controller, described in chapter 3.7.

2.11.8 VME Global Timer

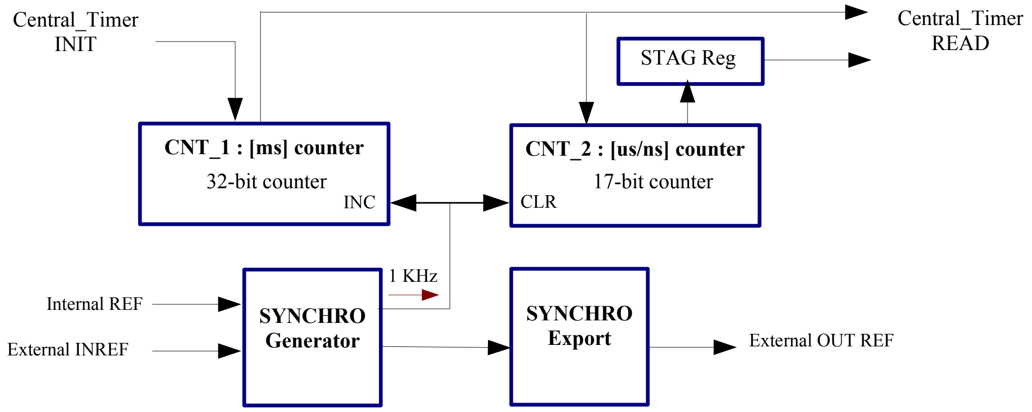
PEV_1100 VME Interface integrates a centralized time base, used for data time-stamping and others time related support. Four registers mapped on the IO_Bus are made available to support the Global Timer service. These registers can be access from the PCI Express IO Space or from the VME CRCSR/A24 decoded area. Refer to chapter 3.3.10

The PEV_1100 Central_Timer is built with two independent counter-timers :

- CNT_1 : 32-bit [ms] counter This counter timer is incremented at every external TOP synchronization reception. (1 KHz). It size allows to built up to ~ 1200 hours without roll-over.
    - Two mode of operation 1000[ms] or 1024[ms] for upward compatibility with TUNDRA Tsi 148 implementation
- CNT_2 : max. 17-bit counter, providing high speed time between the 1[ms] synchronization. This counter can be selected to run at different rate. (1 MHz, 10 MHz, 125 MHz) . This counter timer is clear at every external TOP synchronization reception. (1 KHz) Native precision is derived from the on-board oscillator 50 [ppm] (in 1[ms] maximal drift of +/-50[ns])
- The two counter-timers are mapped in the IO_Bus Space area and can be read from the PCI_Express infrastructure or from the VME64X CRCSR port. To support coherent read-out, a stag register stores the CNT_2 value on primary CNT_1 read-out. CNT_1 can also be loaded at pre-determined value for initialization purpose. (i.e IRIG_B encoded time)

The two counter-timers CNT_1 and CNT_2 are controlled with a 1 KHz TOP synchronization. This synchronization can be selected from following sources :

- Locally built, derived from on-board oscillator (100 MHz 50[ppm]) The 1000 [ms] and 1024[ms] mode is software selectable.
- From VME64x signals (SYSFAIL#, SERCLK# or INT1#/INT2#)
- From external reference (through User's IO on PMC Jn14/Jn24) IRIG_B or GPS or PMC front panel .
- From dedicated user's logic implementing a high stability oscillator (TCXO /VCXO). This could be required if only an external 1PPS information is provided from a GPS equipment. To implement this function a regulation logic (digital PID) shall be designed in the FPGA user's area to control an external VCXO. This kind of logic has been already implemented on IENA system and was providing a 0.05 ppm precision. (less than 50[ns] derating over 1 second)



The local TOP synchronization, (selectable from above described sources) can be broadcast outside the PEV_1100 through following channels

- Over VME signals (SYSFAIL# or SERCLK# or INT1#/INT2#)
- PCI Express
- TBD PCI Express Message

A synchronization monitoring is implemented to survey the CNT_1 and CNT_2 operation. A programmable window (+/- error) is defined to accept/reject the TOP synchronization. Status flags are provided in case of error detection.

The PEV_1100 Global Timer can issues local interrupts derived from the the 1.000[ms] synchronization.

The external synchronization on the VME Bus is supported on SYSFAIL# or with IRQ#1 / IRQ#2. The PEV_1100 can be programmed as Synchronization Master or as Synchronization Slave. Provision is made to support external synchronization (IRIG-B and/or GPS) . FPGA user's area can be assigned to implement such mechanism.

2.11.8.1 IDMA and USER Block Global Time

The VME Global Time is also forwarded to the IDMA and to the USER block.

- The IDMA uses the Global Time information to time-tag the IDMA Descriptor execution (start and end of ...), allowing to trace IDMA execution. It is also used, with specific option to trigger a IDMA channel execution.
- The USER block also receives the Global Time information for user's specific implementation.

2.11.8.2 IRIG_B and GPS Synchronization

The 'true' time is usually distributed through two methods :

- IRIG_B, 1 KHz Amplitude modulated. The 1KHz provides the 1[ms] re-synchronization and the modulation provides 1000 bits informations encoding in BCD the true time (Day, hour, minutes, second). The IRIG-B decoding usually need some analog front-end
- GPS, 1 Hz information 1pps and through a RS232 port the true time information. (Day, hour, minutes, second). Because a synchronization is provided every second, a local high precision oscillator shall be implemented (VCXO + PID + external 1pps)

To support the external Global Time synchronisation, directly by the PEV_1100, an external hardware block (very simple) shall provide following facilities :

- Connection for local TOP synchronization OUTPUT (Multi chassis support). Could be coax SMA/SMB standard connector(s).
- Connection for local TOP synchronization INPUT (Multi chassis support). Could be coax SMA/SMB standard connector(s).
- Connection to IRIG-B. Usually RG58 coaxial connector. An analog front-end (Automatic Gain Amplifier need to be implemented) The IRIG_B frame decoding can be handled locally with FPGA hardware or with firmware running on the local embedded PowerPC440 (If FPGA is FX30T or FX70T). It can also be supported by the HOST
- Connection to GPS Interface, composed of a TBD connector (i.e uDB9) The true time over serial interface (RS232) and the 1PPS synchronization are usually provided over the same cable. Additionally the regulation support (VCXO + DAC) is provided to the PEV_1100 FPGA.

2.11.9 VME P2 IO Port

The PEV_1100 VME P2 user IO (rows A & C) are all wired to the PMC user's IO Jn14 and Jn24.

Signal allocation	VME P2		Signal allocation
PMC #2 Jn24 pin 33	A-1	C-1	PMC #2 Jn24 pin 34
PMC #2 Jn24 pin 35	A-2	C-2	PMC #2 Jn24 pin 36
PMC #2 Jn24 pin 37	A-3	C-3	PMC #2 Jn24 pin 38
PMC #2 Jn24 pin 39	A-4	C-4	PMC #2 Jn24 pin 40
PMC #2 Jn24 pin 41	A-5	C-5	PMC #2 Jn24 pin 42
PMC #2 Jn24 pin 43	A-6	C-6	PMC #2 Jn24 pin 44
PMC #2 Jn24 pin 45	A-7	C-7	PMC #2 Jn24 pin 46
PMC #2 Jn24 pin 47	A-8	C-6	PMC #2 Jn24 pin 48
PMC #2 Jn24 pin 49	A-9	C-9	PMC #2 Jn24 pin 50
PMC #2 Jn24 pin 51	A-10	C-10	PMC #2 Jn24 pin 52
PMC #2 Jn24 pin 53	A-11	C-11	PMC #2 Jn24 pin 54
PMC #2 Jn24 pin 55	A-12	C-12	PMC #2 Jn24 pin 56
PMC #2 Jn24 pin 57	A-13	C-13	PMC #2 Jn24 pin 58
PMC #2 Jn24 pin 59	A-14	C-14	PMC #2 Jn24 pin 60
PMC #2 Jn24 pin 61	A-15	C-15	PMC #2 Jn24 pin 62
PMC #2 Jn24 pin 63	A-16	C-16	PMC #2 Jn24 pin 64
PMC #1 Jn14 pin 33	A-17	C-17	PMC #1 Jn14 pin 34
PMC #1 Jn14 pin 35	A-18	C-18	PMC #1 Jn14 pin 36
PMC #1 Jn14 pin 37	A-19	C-19	PMC #1 Jn14 pin 38
PMC #1 Jn14 pin 39	A-20	C-20	PMC #1 Jn14 pin 40
PMC #1 Jn14 pin 41	A-21	C-21	PMC #1 Jn14 pin 42
PMC #1 Jn14 pin 43	A-22	C-22	PMC #1 Jn14 pin 44
PMC #1 Jn14 pin 45	A-23	C-23	PMC #1 Jn14 pin 46
PMC #1 Jn14 pin 47	A-24	C-24	PMC #1 Jn14 pin 48
PMC #1 Jn14 pin 49	A-25	C-25	PMC #1 Jn14 pin 50
PMC #1 Jn14 pin 51	A-26	C-26	PMC #1 Jn14 pin 523
PMC #1 Jn14 pin 53	A-27	C-27	PMC #1 Jn14 pin 54
PMC #1 Jn14 pin 55	A-28	C-28	PMC #1 Jn14 pin 56
PMC #1 Jn14 pin 57	A-29	C-29	PMC #1 Jn14 pin 58
PMC #1 Jn14 pin 59	A-30	C-30	PMC #1 Jn14 pin 60
PMC #1 Jn14 pin 61	A-31	C-31	PMC #1 Jn14 pin 62
PMC #1 Jn14 pin 63	A-32	C-32	PMC #1 Jn14 pin 64

Table 2.17.: VME P2 A&C User IO assignment

Additionally, on PEV_1100 version equipped with 5-rows connectors, rows D & Z are used to support extra VXI signalling. These side-band signals are connected to PMC #1 Jn13, when populated for VXI extension.

Signal allocation	VME P2		Signal allocation
vxi_LBUS_C11 → (PMC #1 Jn13 pin 16)	D-16	Z-16	GND
vxi_LBUS_C10 → (PMC #1 Jn13 pin 18)	D-17	Z-17	
vxi_LBUS_C9 → (PMC #1 Jn13 pin 22)	D-18	Z-18	GND
vxi_LBUS_C8 → (PMC #1 Jn13 pin 24)	D-19	Z-19	
vxi_LBUS_C7 → (PMC #1 Jn13 pin 28)	D-20	Z-20	GND
vxi_LBUS_C6 → (PMC #1 Jn13 pin 30)	D-21	Z-21	
vxi_LBUS_C5 → (PMC #1 Jn13 pin 34)	D-22	Z-22	GND
vxi_LBUS_C4 → (PMC #1 Jn13 pin 36)	D-23	Z-23	
vxi_LBUS_C3 → (PMC #1 Jn13 pin 40)	D-24	Z-24	GND
vxi_LBUS_C2 → (PMC #1 Jn13 pin 42)	D-25	Z-25	
vxi_LBUS_C1 → (PMC #1 Jn13 pin 46)	D-26	Z-26	GND
vxi_LBUS_C0 → (PMC #1 Jn13 pin 48)	D-27	Z-27	
	D-28	Z-28	GND
vxi_CLK10p → (PMC #1 Jn13 pin 59)	D-29	Z-29	vxiVCC_2Vn → (PMC #1 Jn13 pin 60)
vxi_CLK10n → (PMC #1 Jn13 pin 61)	D-30	Z-30	GND
	D-31	Z-31	vxiVCC_5V2n → (PMC #1 Jn13 pin 64)
	D-32	Z-32	GND

Table 2.18.: VME P2 D&Z VXI IO assignment

2.11.10 VME64X Auto_ID

The PEV_1100 implements the Auto_ID mode to assign the VME64X CR/CSR Base Address. While VME 5-rows connectors are used, the backplane provides a geographic address GA[4:0] + GAP, used to assign the VME64X CR/CSR Base Address. In case of VME 3-rows is used, the Auto_ID method can be an alternative to assign dynamically the VME64X CR/CSR Base Address. While Auto_ID mode is selected, a "Monarch" shall run a specific algorithm, to assign the CR/CSR Base Address of each unit implementing the Auto_ID.

For complete description of the Auto_ID method, refer to VME Specifications Chapter 5.7

To support the Auto_ID mode following hardware support is integrated in the PEV_1100 hardware :

- VME IRQ#2 generation after a VME SYSRESET# deassertion
- Participate on VME IACK level 2
- CR/CSR Base Address writeable from VME Slave interface

To enable the Auto_ID mode, the static switch shall be configured as :

- SW500-3 = "ON" → VME64X CRCSR Mode Enabled
- SW500-[8:3] = ["ON" - "OFF" - "ON" - "OFF" - "ON"] → Magic pattern B"10101" selecting the Auto_ID Mode.
- SW500-[8:3] = ["ON" - "OFF" - "ON" - "OFF" - "OFF"] → Magic pattern B"10100" selecting the Auto_ID Semi-Automatic Mode.

To monitor the Auto_ID configuration process, specific status bits are provided in the “PVME_SLOT1” Register.

[27]	vme64x_Auto_ID	R	-	VME64X Auto_ID Mode enabled. Set while “static_SW_64X = OFF” and SW500-[8:3] = B”1010x”	
[30]	vme64x_RxSYSFAIL	R	-	VME64X SYSFAIL status 0 VME64x SYSFAIL not asserted 1 VME64x SYSFAIL asserted	

[31]	vme64x_Auto_ID_GO	W	0	Command to start Auto_ID Semi-automatic process → IRQ2# assertion + SYSFAIL# de-assertion	
	vme64x_Auto_ID_Semi	R	0	VME64X Auto_ID Semi-Automatic Mode	

The Auto_ID Semi-automatic mode allows to control by SW the Auto_ID process. While enabled, the PEV_1100 asserts VME SYSFAIL# and wait until the “vme64x_Auto_ID_GO” control bit is positioned to '1' to continue the Auto_ID sequence (→ asserts VME IRQ#2 and de-asserts VME SYSFAIL#)

2.11.11 VME64X RMW Transactions

The PEV_1100 is able to issue VME atomic Read Modify Write transactions with CAS (Compare and Swap) logical operation. An independent VME cycle generator, controlled with dedicated registers is instantiated in the VME Master block.

The CAS operation, executed with an atomic RMW cycle is handled with following dedicated registers :

- PVME_RMW_MODE storing general VME informations (AM and OP_Size) used for subsequent RMW operation and the Trigger (start) control.
- PVME_RMW_ADD storing the VME Address used for subsequent RMW operation
- PVME_RMW_CMP storing the compare value. If the VME data read during the RMW match the value, then the VME Write is executed with data stored in the PVME_RMW_UPT register.
- PVME_RMW_UPT storing the update Data.

The CAS operation status (Error, Match_OK or Match_NotOK) is provided in the PVME_RMW_CTL register.

The VME RMW cycle operation uses standard VME arbitration Requester logic, conditioned in the PVME_SLOT1 register. In case of VME RETRY# received during RMW operation, the VME arbitration mastership is released before retrying the RMW operation.

The VME RMW Data register (PVME_RMW_CMP and PVME_RMW_UPT) are implemented with natural VME Data alignment, matching directly the VME Bus DATA lines. For CAS operation, the considered VME Bytes shall be located as defined in the above table.

Size_Mode	**RMW_ADD[1:0]**	**PVME_RMW_CMP / PVME_RMW_UPT**

		[31:24]	[23:16]	[15:8]	[7:0]
D08	00			VME Byte_0	
	01				VME Byte_1
	10			VME Byte_2	
	11				VME Byte_3
D16	0x			VME Byte_0	VME Byte_1
	1x			VME Byte_2	VME Byte_3
D32	00	VME Byte_0	VME Byte_1	VME Byte_2	VME Byte_3

Table 2.19.: VME RMW Data alignment

2.11.12 VME64X CRS Completion

The VME64x specification has defined three transaction acknowledge mechanisms as :

- DTACK → Successful transaction completion.
- BERR → Transaction with Error or Bus time out.
- RETRY → Transaction required to be retried.

In PCI Express environment, only Memory Read transactions, implemented with Read_Request and Read_Completion TLP provide an acknowledge mechanism. Write transactions are allays posted.

The PCI Express Memory Read can be acknowledge with four completion type as :

- Successful Completion (SC)
- Unsupported Request (UR)
- Configuration Request Retry Status (CRS)
- Completer Abort (CA)

In Bridge function, the RETRY acknowledge type can not be reported directly. In PCI Express Specification, Completion TLB with “Compl Status = 010 (CRS)” is only authorized for Configuration Read/Write during the PCI Express enumeration process.

In TOSCA implementation, the VME Master shall be able to issue a RETRY acknowledge with Read_Completion. The RETRY can be implemented with PCI Express field “Compl Status = 010 (CRS)”. This acknowledge mode shall be used only for TOSCA Agent_SW ↔ TOSCA Agent_SW operation. In topology incorporating a PCI Express Switch (as PEV_1100) the Compl Status = 010 (CRS)” can also be used by example for VME -VME Interface built with two PEV_1100.

This mode of operation can be enabled/controlled by setting the control bit field “vmas_CMPL_CRS[1:0]” located in the “PVME_MASCSR” register.

CMPL_CRS	Mode of operation
00	No Completion with Compl. Status = CRS
01	Completion with Compl. Status = CRS on maximum VME RETRY (256) VME RETRY# acknowledge.
10	Completion with Compl. Status = CRS on 1st VME RETRY# acknowledge.

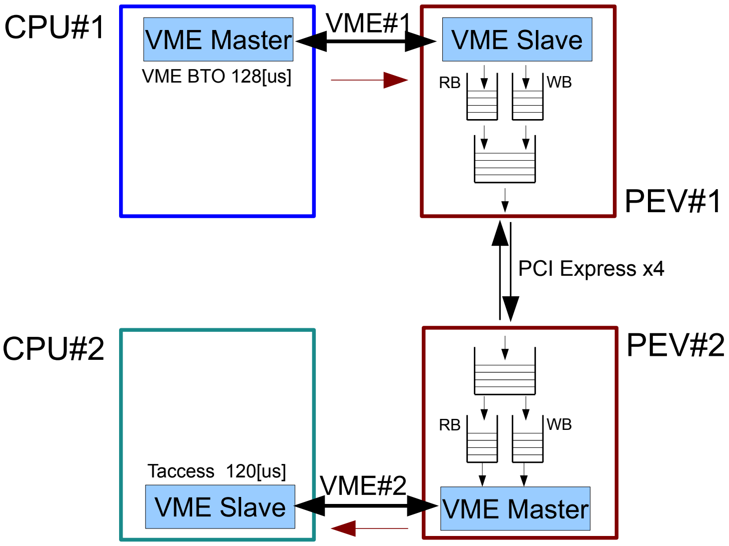
11	Completion with Compl. Status = CRS on fourth VME RETRY# acknowledge.

Additionally for optimal support,the control bit field “vmas_MAXRTCNT[1:0]” located in the “PVME_MASCSR” register provides selectable number of RETRY. (32, 64, 128, or time based)

2.11.13 VME64X Remote RETRY Logic

In system where the PEV_1100 / IPV_1102 are used to interconnect two VME chassis and the remote VME Slave unit exhibit very long access time (> 100[us]), VME Read_Request can be lost due to a Bus Time-out happening is the VME#1 chassis. Following diagram represents the system implementation with system parameters.

- CPU#1 is the Master issuing the VME transaction targeting the CPU#2 (Slave). VME#1 BTO is fixed to 128[us]. (vmeBTO#1)
- CPU#2 is the Slave unit exhibiting very long access time (> 100[us]. VME#1 BTO is fixed to XXX[us]. (vmeBTO#2)
- PEV#1 is acting as VME_Slave to PCIe Initiator Bridge
- PEV#2 is acting as PCIe Target to VME Master Bridge
- PEV units implement segregation on Write_Request Buffer (WB) and Read_Request Buffer (RB). PCI Express interface (EP in PEV and PCIe Switch) implement unified Read_Request/Write_Request Buffer)
- PCI Express guarantee the Write_Request /Read_Request ordering and the PEV's VME Master/Slave interfaces also.



While CPU#1 (VME Master) is writing to CPU#2 (VME Slave) the write transactions are posted. This write posting is implemented across several level of Write Buffer. The reported write transaction in the remote VME#2 is executed uncorrelated from VME#1 operation. In case of very long access time of CPU#2, the posted Write operations are executed continuously until success completion (DTACK# or BERR#) In case of RETRY# received the current posted WRITE is retried indefinitely.

While CPU#1 (VME Master) is reading from CPU#2 (VME Slave) the read transaction is implemented with split transaction. (Read_Request + Read_Completion) The Read_Request is implemented across several level of Read Buffer. The reported read transaction in the remote VME#2 is executed interlocked with VME#1 operation. In case of very long access time of CPU#2, CPU#1 can receive a vmeBTO and terminates its pending Read transaction with a BERR#. → System Error.

This erroneous system behaviour, resulting from a very long VME Slave (CPU#2) access time need to be resolved in critical system. Following workaround is supported by the PEV_1100.

1. A specific logic is incorporated in the VME Slave Controller of PEV#1. It shall be enabled with the control bit “vslv_READ_SPMOD” located in the PVME_SLVCSR register.
2. This logic implement a counter-timer (VSRR_Timer) monitoring the current VME Read transaction. The VSRR_Timer shall be set-up to issue a time-out before the VME#1 BTO. This value can be easily defined as VME#1 BTO value - 1[us]
3. While a VSRR_Timer time-out occurs, the current pending VME Read transaction (issued by CPU#1) is terminated with RETRY# and the PEV#1 VME Slave interface is switched to a BUSY state (... issuing RETRY# on all further decoded VME Read transaction)

   → The VSRC_Timer can be programmed in the PVME_SLVCSR register with control field “vslv_RDRQ_TO[7:0]”.
4. The address information of the VME Read transaction triggering the Read_Request is captured for further match logic support.
5. While the expected Read_Completion is received from the PEV#2, the VME#1 Slave waits for a VME Read transaction matching the previously captured address information. (Refer to item #4) While decoded, the VME Read is acknowledged with the Read_Completion information (Data read + Completion type) The PEV#1 VME Slave interface is released from BUSY state.
6. On VSRR_Timer time-out occurrence → a 2nd counter-timer (VSRC_Timer) is started. The VSRC_Timer time-out value shall be defined to cover maximum response time of any issued Read_Request. This value can be normed and defined as a system parameter (i.e 256 to 2048[us])

   → The VSRC_Timer can be programmed in the PVME_SLVCSR register with control field “vslv_RDRS_TO[1:0]”.
7. While the VSRC_Timer time-out occurs we can consider the issued Read_Request TLP is definitely lost and the Read_Completion will be not received anymore.

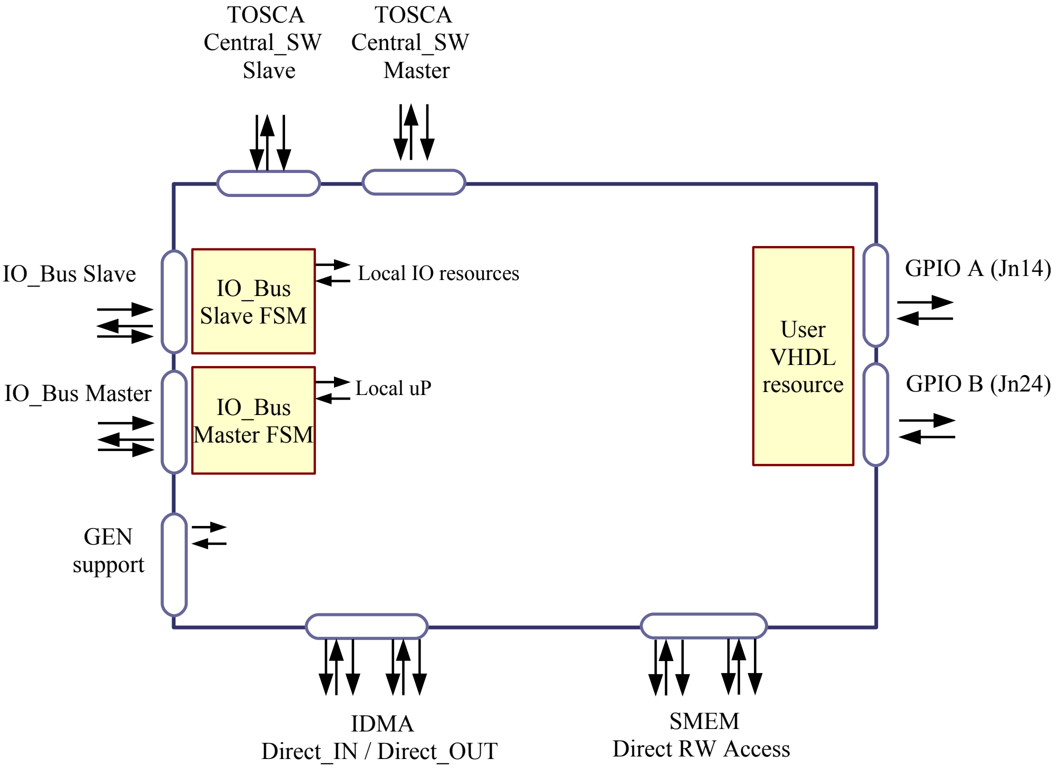
   → The VSRC_Timer time-out flag information is provided in the PVME_SLVCSR register with status bit “vslv_RDRS_TOFL”.
8. The VME Master Controller, active in PEV#2 implement a programmable time based RETRY window in such way that it is not dependant on the number of RETRY received back from the addressed VME Slave. This mode of operation is selectable through control field located in the “PVME_MASCSR” register.

   → The RETRY window time-out mode is enabled with control bit “vmas_MAXRETRY = '0'” and the time value is provided with control field “vmas_MRTY_TIM[7:0]”, both located in the PVME_MASCSR register.

2.12 USER Block

The USER Block is reserved for user specific hardware (VHDL) resource fully embedded in the PEV_1100 infrastructure. PEV_1100 USER block controls 64 GPIO wired to the PMC Jn14/Jn24 user's IO.

The above diagram provides a basic description of the block interfaces.



The USER Block controls 64 GPIO (General Purpose IO), wired to the IEEE1386.1PMC Jn14and Jn24 connectors. Each connectors Jn14 / Jn24 owns 32 signals. User VHDL code can control directly the GPIO signalling and therefore implement the desired function. Next chapter 2.12.2 provides a table with the GPIO pin assignment.

The IO_Bus Slave interfaces the USER Block to the PEV_1100 IO Space infrastructure. A 1 KBytes area is reserved for it. Refer to chapter 3.1.1 for IO_Bus Slave description. The interrupt generation to the Host controller is also supported through IO_Bus Slave side band signalling.

The IO_Bus Master interface allow local intelligence embedded in the USER Block to get full access to the IO_Bus resources. Local intelligence can be embedded processor as PowerPC440 or soft IPcore as MicroBlaze. Refer to chapter 3.1.2 for IO_Bus Slave description.

The GEN support provide side band signalling as CLOCK reference, RESET control, Global Time, ...

The Central_SW Slave port allows to map the USER Block resources directly to the MEMORY Space. The TOSCA Central_SW infrastructure implements PCI Express TLP READ and WRITE with burst capability support up to 512 Bytes. The interface supports up to 2* 1.6 GBytes/s (simultaneous write and read) Following three transactions type are supported :

- Write_Posting (PCI Express Write Memory Request)

- Read_Request (PCI Express Read Memory Request)
- Read_Response (PCI Express Read Completion)

The Central_SW Master port allows local intelligence (processor or dedicated DMA controller) to have direct access to the TOSCA Central_SW infrastructure. The Master shall be able to manage PCI Express TLP with the adequate format. (Write_Posting, Read_Request, Read_Response) The interface supports up to 2* 1.6 GBytes/s (simultaneous write and read).

The IDMA Direct IN/OUT ports are directly attached to the PEV_1100 IDMA controller. Refer to chapter 2.10. The IDMA engines can interface directly IO channels through these private and simple interfaces without using the Central_SW infrastructure. This mode of operation is described in chapter 2.10.3. The Direct IN port is associated with the WRITE_Engine #0 and the Direct OUT port is associated with the READ_Engine #0

The SMEM Direct access port provides direct path to the SMEM DDR2 multi-port controller. Two independent interfaces are supplied for independent Read and Write. This SMEM Direct access port have simpler protocol and better latency control then access across the TOSCA Central_SW.

In current implementation, the USER block integrates following function :

- GPIO Programmable control
- MEMORY space to IO_Bus Bridge
- Four(4) Message Passing FIFO (255x32)

2.12.1 TOSCA FPGA Design Kit

The TOSCA “FPGA Design Kit” provides a complete design environment to support the USER Block implementation.

On current implementation the TOSCA “FPGA Design Kit” is not supported

2.12.2 GPIO Programmable control

The PEV_1100 implements a default USER block providing direct control of the 64 available GPIO signals connected to the PMC Jn14/Jn24 User IO. Two set of 32 GPIO are defined as “GPIO_A” and “GPIO_B”.

Each GPIO can be programmed to act as OUTPUT or INPUT only with “GPIO_A_ENA / GPIO_B_ENA”, “GPIO_A_OUT / GPIO_B_OUT and “GPIO_A_IN / GPIO_B_IN” registers .

Above table sums up the IO assignation.

PMC Signal allocation		Pin#		PMC Signal allocation	
Jn24 (Top)	Jn14 (Bottom)			Jn14 (Bottom)	Jn24 (Top)
fpga_GPIO_1	fpga_GPIO_33	1	2	fpga_GPIO_35	fpga_GPIO_3
fpga_GPIO_2	fpga_GPIO_34	3	4	fpga_GPIO_36	fpga_GPIO_4
fpga_GPIO_5	fpga_GPIO_37	5	6	fpga_GPIO_ 39	fpga_GPIO_7
fpga_GPIO_6	fpga_GPIO_38	7	8	fpga_GPIO_40	fpga_GPIO_8
fpga_GPIO_9	fpga_GPIO_41	9	10	fpga_GPIO_ 43	fpga_GPIO_11

fpga_GPIO_10	fpga_GPIO_42	11	12	fpga_GPIO_44	fpga_GPIO_12
fpga_GPIO_13	fpga_GPIO_45	13	14	fpga_GPIO_47	fpga_GPIO_15
fpga_GPIO_14	fpga_GPIO_46	15	16	fpga_GPIO_48	fpga_GPIO_16
fpga_GPIO_17	fpga_GPIO_49	17	18	fpga_GPIO_51	fpga_GPIO_19
fpga_GPIO_18	fpga_GPIO_50	19	20	fpga_GPIO_52	fpga_GPIO_20
fpga_GPIO_21	fpga_GPIO_53	21	22	fpga_GPIO_55	fpga_GPIO_23
fpga_GPIO_22	fpga_GPIO_54	23	24	fpga_GPIO_56	fpga_GPIO_24
fpga_GPIO_25	fpga_GPIO_57	25	26	fpga_GPIO_59	fpga_GPIO_27
fpga_GPIO_26	fpga_GPIO_58	27	28	fpga_GPIO_60	fpga_GPIO_28
fpga_GPIO_29	fpga_GPIO_61	29	30	fpga_GPIO_63	fpga_GPIO_31
fpga_GPIO_30	fpga_GPIO_62	31	32	fpga_GPIO_564	fpga_GPIO_32

Table 2.20.: USER Block GPIO - IEEE 1386.1 Jn14/Jn24

The signals GPIO_A_IN[1:0] and GPIO_B_IN[3:0] are directly connected to the USER Agent_SW ITC16. The interrupt polarity can be chosen through control registers "GPIO_A_POL" and "GPIO_B_POL" .

2.12.3 IO_Bus Bridge

The IO_Bus Bridge function is implemented in the USER block, with on the fly protocol translation. It can be accessed from any of the Tosca Agent_SW Master (PCIe_EP, VME64x or IDMA) This function is particularly useful while the IO_Bus resources shall be accessible from PCI Express MEMORY Space. (I.e access from PCI Express Non Transparent Bridge)

The NoPF INgress MMU Page #3 (NoPF Base + 3 MBytes is pre-initialized to map diretcly the USER Agent_SW implementing the IO_Bus Bridge)

The IO_Bus Bridge function is implemented with two main blocks as :

1) Tosca MEMORY Slave Interface, directly connected to the Central Switch handling following TLP packet
   - Write_Posted TLP
   - Read_Request and Read_Completion TLP
2) IO_Bus Master Interface, running at 125 MHz, and generating IO_Bus Read and IO_Bus Write transactions. The IO_Bus owns a central arbiter managing the multiple access sharing.

The IO_Bus Bridge supports only single D32 word transaction. D64 or Block transfer transaction are not supported.

The IO_Bus Bridge function is located in the high section of the first 1 MBytes addressing area. Provision is made to support Tosca II implementation with up to eight(8) Agent_SW.

Following table sums up the USER Space address decoding.

Local USER AgentSW ADD[31:0]	Implementation
0x0010'0000 - 0xFFFF'FFFF	Reserved Not supported
0x000F'F000 - 0x000F'FFFF	Reserved for extended IO_Bus (Tosca II Implementation)

<table>
<tr><td><code>0x000F'E000 - 0x000F'EFFF</code></td><td>IO_Bus Bridge function
<table>
<tr><td><code>000F'E000 - 000F'E3FF</code></td><td>PCIe_EP/ LOCAL Agent_SW</td></tr>
<tr><td><code>000F'E400 - 000F'E7FF</code></td><td>VME64X Agent_SW</td></tr>
<tr><td><code>000F'E800 - 000F'EBFF</code></td><td>SMEM_IDMA Agent_SW</td></tr>
<tr><td><code>000F'EC00 - 000F'EFFF</code></td><td>USER Agent_SW</td></tr>
</table></td></tr>
<tr><td><code>0x0000'0010 - 0x000E'FFFF</code></td><td>Not used.. Access to this area cause transaction Error</td></tr>
<tr><td><code>0x0000'0000 - 0x0000'000F</code></td><td>Four (4) Message Passing FIFO addressing area (Refer to chapter 2.5)</td></tr>
</table>

Table 2.21.: IO_Bus Bridge MEMORY Mapping

Because the IO_Bus Bridge function in mapped over PCI Express MEMORY space, the Write transactions are posted and therefore can not report directly error cases. A IO_Bus Bridge write-posting error Interrupt is issued on the ITC16 INT#15 Refer to chapter 2.5

The Read transaction can report error through appropriate Read_Completion TLP format.

2.12.4 Message Passing FIFO

The four Message Passing FIFO are implemented in the USER block. They are implemented with a RAM storage organized in 4x256x32 and a dedicated control logic handling FIFO sequence running at 125 MHz.

- FIFO size = 255 x 32 bit. (All four implemented in a single 36K BlockRAM)
- Flags FULL and NotEMPTY connected to dedicated ITC16
- Embedded control logic handling the FIFO management

The flags FULL and NotEMPTY of each FIFO are updated in real-time and supplied to the USER Agent_SW Interrupt Controller ITC16. Each individual flag can be programmed to issue a PCI Express MSI interrupt.

The Message Passing FIFO Read/Write Ports are mapped through two independent access space. Both can be used at same time thanks to an embedded arbitration controlling sharing the access from the two ports.

1. IO_Bus. Access through the IO_Bus locations a
2. TOSCA MEMORY switched path. Access through this path allows to map the Message Passing FIFO on the VME A32 Slave. To achieve it, a dedicated 1 MBytes page shall be programmed in VME INgress MMU table. Access through the MEMORY switched path can be protected in Write and Read for each FIFO with two control bits:
    - “MSG_FIFO_TMEM_REA” → Read FIFO#n Enable
    - “MSG_FIFO_TMEM_WEA” → Write FIFO#n Enable

Only D32 transactions are supported for the Message Passing FIFO. D64 and block transfer are rejected.

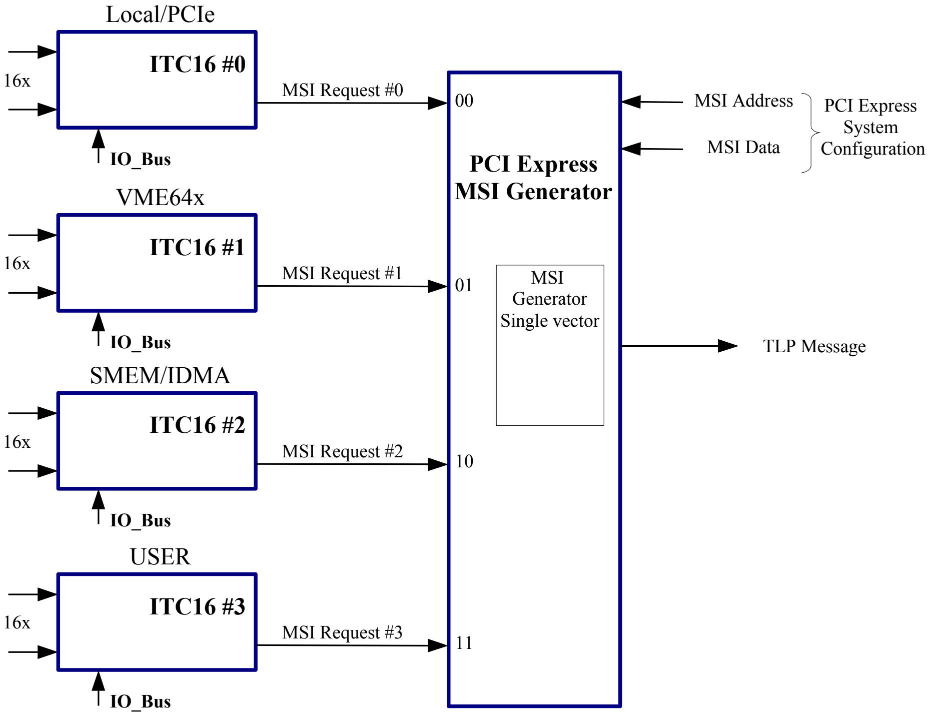
Each Message Passing FIFO is controlled with a dedicated control & status register. These four registers control and monitor the Message Passing FIFO These registers are only mapped over the IO_Bus path. Refer to chapter 3.6.9 and 3.6.10

2.13 Interrupt Infrastructure

The PEV_1100 implements a complete Interrupt infrastructure dispatched to the PCI Express Host Controller. The PCI Express MSI generator, manages the MSI generation, on request from the attached ITC16.

The PEV_1100 instantiates four ITC16 blocks providing management for up to 64 local interrupts. The four ITC16 issues a unique PCI Express MSI. The PCI Express MSI parameters are defined through the PCI Express System Configuration. (Refer to PCI Express specification)

Due to LINUX limitation, only single vector MSI is supported. To work-around this system integration issue, a dedicated hardware FSM has been implemented to support the four on chip ITC serviced to a single MSI message.



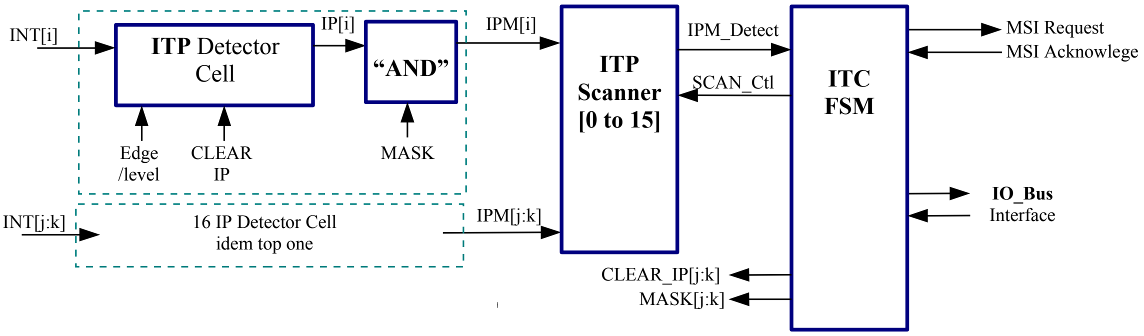
The four MSI Request, issued by the ITC16 are handled sequentially in a round robin arbitration mode. The PCI Express MSI generator block acts as :

1. Generate the PCI Express MSI Message on detection of one of the local MSI Request (#0, #1, #2, #3). At the MSI message generation, the MSI Request source is memorized for expected Interrupt Acknowledge sequence.
2. The dedicated logic wait for a “ITC#0 IACK_REG” read access Refer to chapter 3.7.1. This read cycle will provide a 16-bit vector signifying the interrupt source. The dedicated logic routes automatically the read access to the serviced MSI Request .
3. The dedicated logic wait for a “ITC#0 IACK_REG” write access. Refer to chapter 3.7.1 This write access will clear the ITP in the currently serviced ITC and re-enable the corresponding ITC. The logic route the write access to the serviced MSI Request.

2.13.1 ITC16 Interrupt Controller

The PEV_1100 interrupt is based on a normalized 16 IRQ Interrupt Controller block. This block in instantiated in each Agent_SW block. The ITC16 integrates following features :

- Edge or Level hardware selectable
- Individual Mask programmable with independent SET & CLEAR command register
- Interrupt Acknowledge Register with autonomous clear of Interrupt Pending
- VME automatic IACK pre-fetching



The 16 interrupt sources are primary handled by the "ITP Detector Cell", generating Interrupt Pending signal. (ITP) The Mode of operation Edge/Level is fixed by hardware for each of the source. While an ITP is handled by the Host, the ITC FSM automatically clears it.

The ITP can be masked individually with specific control bit located in the Mask register. This function is controlled with the "ITC_IMC" and "ITC_IMS" register, providing independent Bit SET and Bit CLEAR operation. Refer to chapter 3.7.3 and 3.7.4.

A 16-bit multiplexer associated with a 4-bit counter implements the "ITP Scanner" function. This scan logic is completely under control of the ITC FSM. The "ITP Detector Cell"are continuously monitored until an ITP is detected.

When an ITP is detected active, the scan is stopped and a MSI_Request is send to the PCI Express MSI logic. While the MSI is effectively send to the Host, with specific PCI Express TLP, a MSI_Acknowledge is send back to the ITC FSM.

After a MSI_Request/MSI_Acknowledge sequence , the ITC_FSM is expecting an ITC_IACK read cycle. Refer to chapter 3.7.1 This IO_Bus read provides a 16-bit identifier to the Host Interrupt Handler, allowing to identify the ITP source.

The Host Interrupt Handler has the responsibility to clear the ITP (Interrupt Pending) and to restart the scan. This action is made by writing to the "ITC_IACK" Register with selected IP marked to be cleared. Refer to chapter 3.7.1

The following table sums up the four PEV1100 ITC16 interrupt sources

INT#	ITC16 Local	ITC16 VME64x	ITC16 SMEMIDMA	ITC16 USER
#0	V5_SMON_ALM[0]	VME SYSRESET# ass.	IDMA RDengine#0 END	MSG_FIFO#0_NotEMPTY
#1	V5_SMON_ALM[1]	VME IRQ_1	IDMA RDengine#0 ERR	MSG_FIFO#1_NotEMPTY
#2	V5_SMON_ALM[2]	VME IRQ_2	IDMA RDengine#1 END	MSG_FIFO#2_NotEMPTY
#3	V5_SMON_OT System	VME IRQ_3	IDMA RDengine#1 ERR	MSG_FIFO#3_NotEMPTY
#4	LM86 CRIT	VME IRQ_4	IDMA WRengine#0 END	MSG_FIFO#0_FULL

#5	LM86 THERM Alarm	VME IRQ_5	IDMA WRengine#0 ERR	MSG_FIFO#1_FULL
#6	Not used	VME IRQ_6	IDMA WRengine#1 END	MSG_FIFO#2_FULL
#7	Not used	VME IRQ_7	IDMA WRengine#1 ERR	MSG_FIFO#3_FULL
#8	Not used	VME ACFAIL#	Not used	GPIO_A_0
#9	Not used	VME_Error (Master)	Not used	GPIO_A_1
#10	Not used	VME Central Time #1	Not used	GPIO_B_0
#11	Not used	VME Central Time #2	Not used	GPIO_B_1
#12	Not used	Location Monitor #0	Not used	GPIO_B_2
#13	Not used	Location Monitor #1	Not used	GPIO_B_3
#14	Not used	Location Monitor #2	Not used	MSG FIFO WP Error
#15	Not used	Location Monitor #3	Not used	IO_Bus Bridge WP Error

Table 2.22.: PEV_1100 ITC16 INT sources

Before using the Interrupt logic, the four ITC shall be enabled and selected IT sources shall be unmasked. Refer to chapters 3.7.3 and 3.7.4.

2.14 Environmental

The PEV_1100 is designed for commercial temperature grade . The above table sums up the environmental.

	Commercial Grade	Comments
Operational Temperature	0 to 50 [ºC]	With 400 LFM forced air cooling.
Storage Temperature	- 55 to 105 [ºC]	
Relative Humidity	5% to 95%	Non condensing
Vibration	To Be Defined	
Shocks	To Be Defined	
Altitude	0 – 5000 meters	
EMI / RFI	EN50022-Class A EN50082	
Security	EN60950	

Table 2.23.: PEV_1100 Environmental

Due to the difficulties to get precise informations on the air cooling efficiency and the installed PMC/XMC, the PEV_1100 provides on-board thermal sensors.

- Internal XILINX Virtex-5T die temperature monitor. Refer to chapter 2.5.3
- External LM86 temperature sensor.

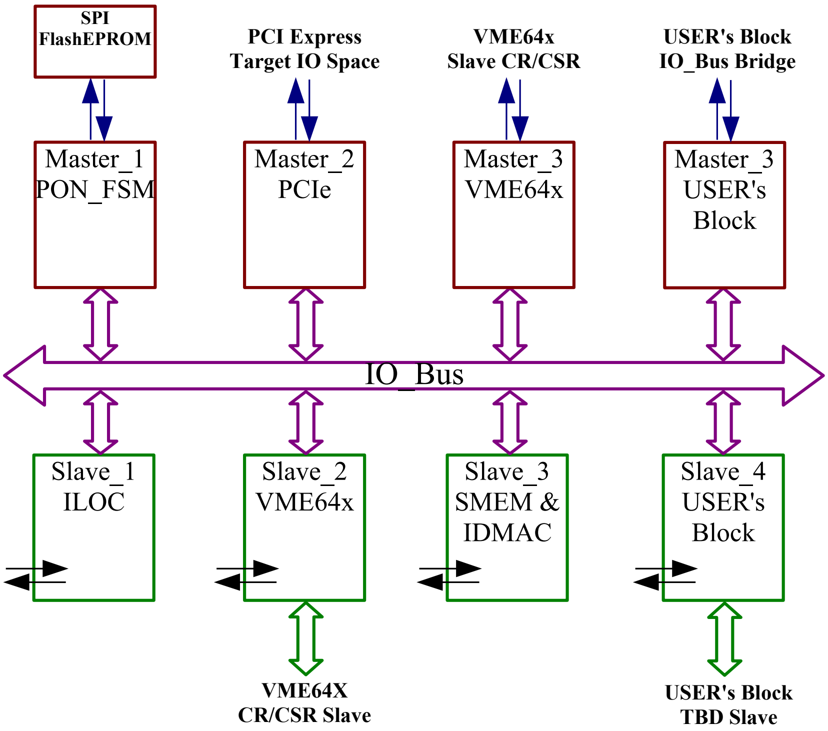
These two thermal informations allow to characterize the thermal envelope for the specific integration.

3 IO_Bus Control & Status Register

The following chapter provides complete PEV_1100 IO Space description.

3.1 IO_Bus Infrastructure

The IO_Bus Infrastructure allows to interface the PEV_1100 internal registers and resources with the potential on-board controller. This multi Master shared media is implemented with a 8-bit Address/Data arbitrated multiplexed Bus running at 125 MHz.



The above diagram represents the PEV_1100 IO_Bus infrastructure, interconnecting the four potential Master with four addressed Slaves. The IO_Bus support only single beat Read and Write transactions.

3.1.1 IO_Bus Master

The PV-1100 IO_Bus Infrastructure supports four potential IO_Bus Master :

1. PON_FSM, Autonomous FSM Sequencer started at power-up, allowing to issues IO_Bus Write, directly fetched from the on-board SPI Flash EPROM. Up to 65K instruction can be stored in the SPI Flash EPROM and executed sequentially before releasing the internal RESET. The PON_FSM allows to initialize completely the PEV_1100 without any external software boot process. Refer to chapter 3.1.3
2. PCI_Express port. The IO_Bus is fully mapped in the PCI_Express IO Space. This occupies a total of 4 KBytes or 256 Bytes (compressed mode) area in the PCI Express IO Space,

set-up by the appropriate BAR(x) configuration register.

3. VME64x port. The IO_Space is fully mapped in the VME64x CR/CSR area. The PEV_1100's internal resources can be fully addressable from the VME64x Bus.
4. USER port. This section incorporates a MEMORY to IO_Bus Bridge function providing IO_Bus mapped resources over the MEMORY space.

3.1.2 IO_Bus Slave

The PEV-1100 IO_Bus infrastructure maps four IO_Bus Slave. Each IO_Bus Slave occupies a fixed 1 KBytes space (256* D32 registers). Some of the IO_Bus Slave also incorporates a 2nd local Bus access. This 2nd port access is arbitrated internally with the IO_Bus access.

1. ILOC, PCI Express EP and Local CSR Resources. This IO_Bus Slave owns all PEV_1100 general registers resources. Refer to Chapter 2.2.
2. VME64X. This IO_Bus Slave owns the VME64x CR/CSR Configuration resources. The concerned resources are also directly mapped with the VME Slave CR/CSR controller, providing a low latency path without potential dead-lock.
3. Shared Memory & IDMAC. This IO_Bus Slave owns all related registers controlling the DDR2 Shared Memory and IDMAC.
4. USER Block. This IO_Bus Slave owns related registers and resources for the GPIO control and the Message Passing FIFO

3.1.3 PONFSM Controller

The PON_FSM is an autonomous sequencer involved at power up. If not disable by SW502-2, This IO_Master controller executes a start-up microcode stored in the SPI Serial FlashEPROM.

The PON_FSM microcode is stored on the latest 128 KBytes area (1 Mbit) of concerned bit-stream, supporting execution of up to 16 K contiguous microcode instructions. All microcode instructions are executed in a fixed 1.2[us].

Each PON_FSM Instruction is built within a 64-bit word, occupying eight consecutive Bytes in the SPI Serial FlashEPROM.

Three basic PON_FSM Instructions are implemented as :

1. IOBUS_WRITE, allowing to initiate IO_Bus Write transactions and therefore initialize all PEV_1100 internal resources mapped over the IO_Bus.
2. WAIT, allowing to implements a fixed wait
3. STOP, terminating the FSM_PON execution .

Following table provides the PON_FSM instruction format :

DWORD_1 Instruction Coding			
[31:24]	Instruction Code	8-bit field encoding the Instruction Code. 0x45 = IOBUS_WRITE 0x34 = WAIT 0x22 = STOP Others = ILLEGAL	Only code 0x45, 0x34 and 0x22 are interpreted as valid PON_FSM Instructions. Any other code stops immediately the FSM_PON execution with illegal case.
[23:16]	Instruction Reserved	Not used	
[15:0]	Instruction Extension	16-bit field specifying the Instruction Extension. IO_Bus Address or Wait time in 1[us] step. (up to 65[ms])	The IO_Bus Address is encoded as follow : [1:0] : Not used. [9:2] : IO_Bus Slave resource selection in LWORD. [13:12] : IO_Bus number selection. (0,1,2,3) [15:14] : Reserved
DWORD_2 Associated Data			
[31:0]	Instruction Data	32-bit field used to IOBUS_WRITE	Natural 32-bit LWORD format. Not used for WAIT Instruction.

Table 3.1.: PONFSM Instruction Format

A specific design environment is provided to built the PON_FSM microcode and to load it in the SPI Serial Flash EPROM with concerned FPGA bit-stream. Refer to chapter 2.5.2

The PON_FSM execution at power-up can be disabled by setting SW502-2 “ON”.

The PON_FSM execution provides status informations in the “ILOC_PONFSM” register. Refer chapter 3.2.4

3.1.4 PCIe MEMORY Bridge

The PEV_1100 integrates in the USER Agent_SW a transparent PCIe MEMORY to IO_Bus Bridge function. This function is particularly useful while the IO_Bus resources shall be accessible from PCI Express MEMORY Space. (I.e access from PCI Express Non Transparent Bridge)

The NoPF INgress MMU Page #3 (NoPF Base + 3 MBytes is pre-initialized to map diretcly the USER Agent_SW implementing the IO_Bus Bridge)

Refer to Chapter 2.12.13

3.2 Local CSR Resources

The LOCAL CSR Resources are mapped on a single-port interface

- IO_Bus 1 KBytes Device#1 (0x000-0x3FF)

The LOCAL registers provide the facilities to control the PCI Express Endpoint Interface. The following table sums-up the LOCAL register mapping.

Offset IBUS	Size	Register_Name	Description	Comments
0x000 - 0x003	0x4	**ILOC_STATIC**	External Static Information (micro-switch & strapping)	
0x004 - 0x007	0x4	**ILOC_CABLE_1**	External PCI Express Connector #1	
0x008 - 0x00B	0x4	**ILOC_CABLE_2**	External PCI Express Connector #2	
0x00C - 0x00F	0x4	**ILOC_PONFSM**	Power_ON FSM Sequencer Status	
0x010 - 0x013	0x4	**ILOC_SPI**	SPI Flash EPROM Bit programming	
0x014 - 0x017	0x4	**ILOC_PCIE_SW**	PCI Express Switch PEX8624 Control & Status	
0x018 - 0x01B	0x4	**ILOC_SIGN**	Signature FPGA date of Creation 32-bit field	
0x01C - 0x01F	0x4	**ILOC_GENCTL**	General Control Register	
0x020 - 0x023	0x4	**PCIE_MMUADD**	PCI Express INgress MMU Address pointer.	
0x024 - 0x027	0x4	**PCIE_MMUDAT**	PCI Express INgress MMU Data Register	
0x028 - 0x02B	0x4	**PCIE_EPSTA**	PCI Express EP Status	
0x02C - 0x03F	0x14	**ILOC_Reserved**	Not used	
0x040 - 0x43	0x4	**V5_SMON_ADDPT**	VIRTEX-5 System Monitor Address Pointer	
0x044 - 0x47	0x4	**V5_SMON_DAT**	VIRTEX-5 System Monitor Data Register	
0x048 - 0x4B	0x4	**V5_SMON_STA**	VIRTEX-5 System Monitor Status	
0x04C - 0x5F	0x14	**V5_SMON_Reserved**		
0x060 - 0x63	0x4	**V5_PCIEP_ADDPT**	VIRTEX-5 PCI Express EP Address Pointer	
0x064 - 0x67	0x4	**V5_PCIEP_DAT**	VIRTEX-5 PCI Express EP Data Register	
0x68 - 0x6B	0x4	**V5_PCIEP_SEL**	VIRTEX-5 PCI Express EP Status Selection	
0x6C - 0x6F	0x4	**V5_PCIEP_RSLT**	VIRTEX-5 PCI Express EP Selection Result	
0x070 - 0x7F	x0F	**V5_PCIEP_Reserved**		
0x080 - 0x083	0x4	**PCIE_BE_ITC_IP**	PCIE_BE ITC Interrupt Pending	
0x084 - 0x087	0x4	**PCIE_BE_ITC_IACK**	PCIE_BE Local ITC Interrupt Acknowledge	
0x088 - 0x08B	0x4	**PCIE_BE_ITC_IMC**	PCIE_BE ITC Mask CLEAR Control Register	
0x08C - 0x08F	0x4	**PCIE_BE_ITC_IMS**	PCIE_BE ITC Mask SET Control Register	
0x0C0 - 0x0C3	0x04	**PMCXMC**	PMC & XMC Side-band Signalling	

Offset IBUS	Size	Register_Name	Description	Comments
0x0C4 - 0x0FF	0x3C	**Reserved**		
0x0100 - 0x0103	0x4	**I2C_CTL**	I2C Controller Control Register	
0x0104 - 0x0107	0x4	**I2C_CMD**	I2C Controller Command Register	
0x0108 - 0x010B	0x4	**I2C_DATW**	I2C Controller Data Write Register	
0x010C - 0x010F	0x4	**I2C_DATR**	I2C Controller Data Read Register	
0x380 - 0x3FF	0x80	**Signature_ROM**	Defined ROM area for software signature	

3.2.1 ILOC_Static Register

This register provides status informations related to on-board static options. These status are provided from the PYRAME CPLD over a serial stream.

ILOC_STATIC IO_Bus : 0x000 - 0x003					
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[0]	SW500-1	R	--	VME SYSRESET Enable	
[1]	SW500-2	R	--	VME Slot_1 Enable (Force anyway)	
[2]	SW500-3	R	--	VME64X Mode Enable	
[3]	SW500-4	R	--	VME A24 Mode A[19]	
[4]	SW500-5	R	--	VME A24 Mode A[20]	
[5]	SW500-6	R	--	VME A24 Mode A[21]	
[6]	SW500-7	R	--	VME A24 Mode A[22]	
[7]	SW500-8	R	--	VME A24 Mode A[23]	
[8]	SW501-1	R	--	SPI Flash Programming Enable	
[9]	SW501-2	R	--	FPGA Bit-stream #N LSB	
[10]	SW501-3	R	--	FPGA Bit-stream #N MSB	
[11]	SW501-4	R	--	Reserved_1	
[12]	SW501-5	R	--	PCI Express Front panel P4 Non Transparent	
[13]	SW501-6	R	--	PCI Express Mode [2:0] LSB	
[14]	SW501-7	R	--	PCI Express Mode [2:0]	
[15]	SW501-8	R	--	PCI Express Mode [2:0] MSB	
[16]	SW502-1	R	--	FPGA Configuration made with XILINX iMPACT JTAG tool.	
[17]	SW502-2	R	--	Compressed 256 Bytes IO Space	
[18]	SW502-3	R		PCI Express BAR_2 Size (from 64 to 256 MBytes + 4 Mbytes) (Non Prefetchable)	
[19]	SW502-4	R	--		
[20]	SW502-5	R	--	PCI Express BAR_0 Size (from 64 to 4096 MBytes) (Prefetchable)	
[21]	SW502-6	R	--		
[22]	SW502-7	R	--		

ILOC_STATIC IO_Bus : 0x000 - 0x003					
[23]	SW502-8	R	--	PCI Express BAR_0 64-bit	
[25:24]	Dyn_IOSEL[1:0]	RW	b00	256 bytes Compress IO Space Dynamic section selector (Refer to annexe X.X)	
[30:26]	Reserved	R	--	Not used	
[31]	Compressed_IO	R		This status information is set while the Compressed 256 bytes IO_Space is enabled.	Idem as SW502-2

Note *Switch numbering SW50X.**K** corresponds to the mini-DIP switch number printed label (1 to 8)*

Note *The HEX rotary switch is connected on pex_GPIO[3:0]. Refer to register ILOC_PCIE_SW.*

3.2.2 ILOC_Cable_1 [P4] Register

This register provides control & status informations related to the PCI Express External Cabling Port_1 Connector P4. Connected to PEX-8624-AA Port 1 .

ILOC_Cable_1 IO_Bus : 0x004 - 0x007	PCI Express External Cabling P4				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[2:0]	Reserved	R	--	Not implemented	
[3]	Port1_Non Transparent	R	--	Port_1 as NT (Non Transparent)	Not supported in normal operation
[4]	Port1_CPERST_I	R	--	Platform RESET Detection	
[5]	Port1_CPWRON_I	R	--	Cable Power_ON Detection	
[6]	Port1_CWAKE_O	RW	0	Signalling Wake-Up over the PCI Express Cable	
[31:7]	Reserved	R	--	Not implemented	

3.2.3 ILOC_Cable_0 [P3] Register

This register provides control & status informations related to the PCI Express External Cabling Port_2 Connector P3. Connected to PEX-8624-AA Port 0.

ILOC_Cable_0 IO_Bus : 0x008 - 0x00B	PCI Express External Cabling P3				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[3:0]	Reserved	R	--	Not implemented	
[4]	Port2_CPERST_I	R	--	Platform RESET Detection	
[5]	Port2_CPWRON_I	R	--	Cable Power_ON Detection	
[6]	Port2_CWAKE_O	RW	0	Signalling Wake-Up over the PCI Express Cable	
[31:7]					

3.2.4 ILOC_PONFSM Register

This register provides status informations related to Power-ON Sequencer. The PON_FSM Controller executes a autonomously at power-up a predefined sequence stored in the SPI Flash EPROM.

ILOC_PONFSM IO_Bus : 0x00C - 0x00F	PCI PONFSM Sequencer				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[15:0]	PONFSM_INSTR[15:0]	R	0x0000	Number of PON_FSM Instruction executed	
[17:16]	PONFSM_STA[1:0]	R	0b00	PON_FSM Execution End Status 00 = Not Started. 01 = Running. 10 = End OK. 11 = End with ERROR. (Aborted)	
[18]	Reserved	R	0	Not used	
[19]	PONFSM_Enable	R	0	PON FSM is not by-passed)	
[31:20]	PONFSM_SIGNREV	RW	0x000	12-bit Register reserved for PON_FSM signature execution. Write only authorized by the PON_FSM providing PON_FSM microcode tracing	

3.2.5 ILOC_SPI Register

This register provides control &status information to support on-board SPI Flash EPROM programming. The SPI interface protocol shall be completely controlled under software.

ILOC_SPI IO_Bus : 0x010 - 0x013	SPI Flash EPROM programming				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[0]	pgm_SPICLK	W	'0'	SPI Clock calibrated pulse. While set a calibrated 32 [ns] pulse is automatically generated, with controlled set-up /hold time related to pgm_SPIDO	Need time accurate sequencing.
[1]	pgm_SPIDO	RW	'0'	SPI Data OUT direct pin control	
[2]	pgm_SPIDI	R	'0'	SPI Data IN direct pin status	
[3]	pgm_SPICS[0]	RW	'0'	SPI Chip Select direct pin control. '1' for CS active.	
[4]	pgm_REMCMD	RW	'0'	Remote RESET/FPGA bit-stream reload commands. While activated, to '1', the FPGA is re-configured with bit-stream number selected with [pgm_SPIDO: pgm_SPICLK] If (pgm_REMCMD= '1' and pgm_SPICS[0]= '1') → [pgm_SPIDO , pgm_SPICLK] : = 00 : No action = 01 : Remote RESET = 10 : No action = 11 : VME SYSRESET If (pgm_REMCMD= '1' and pgm_SPICS[0]= '0') →	0Can also be used to validate new FPGA bit-stream. Remote RESET CMD FPGA Reload

ILOC_SPI IO_Bus : 0x010 - 0x013		SPI Flash EPROM programming			
				[pgm_SPIDO , pgm_SPICLK] : = 00 : FPGA reload Bit-stream 0 = 01 : FPGA reload Bit-stream 1 = 10 : FPGA reload Bit-stream 2 = 11 : FPGA reload Bit-stream 3	
[5]	pgm_REMFPGA	RW	'0'	Enable remote FPGA Configuration reload	
[31:6]	Reserved	R	--	Not implemented	

Note Remote RESET with FPGA reload will break the PCI Express communication. A complete re-initialisation process shall be executed on the Host and /or on the PCI Express switch PEX 8624-AB.

3.2.6 ILOC_PCIE_Switch Register

This register provides status informations dedicated to the external PCI Express Switch PEX 8624-AA. For complete technical information refer to the PLX PEX8624 Technical Documentation.

ILOC_PCIE_SW IO_Bus : 0x014 - 0x017		PCI Express Switch PEX8624			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[3:0]	pex_UP_Select	R	0x0	PCI Express Switch Port defined as UP_stream	Read-only condition
[5:4]	pex_NT_Select	R	0x0	Micro switch PCI Express Switch "NT" conditioning. Refer to static option chapter	Read-only condition
[6]	pex_NT_Enable	R	0x0		Read-only condition
[7]	pex_PERST	R	0b1	PCI Express RESET command	Read-only condition
[8]	pex_FATAL_ERR	R	--	Direct PEX8624 status information	Refer to PLX PEX8624AB technical manual.
[9]	pex_INTA	R	--	Direct PEX8624 status information	
[10]	pex_NTRESET	R	--	Direct PEX8624 status information	
[11]	Reserved	R	--	Not implemented	
[13:12]	pex_TYPE			Strap status providing information on PLX Switch type populated [00] = PEX8612 : 3 PCIEx4 ports [01] = PEX8616 : 4 PCIEx4 ports [10] = PEX8624 : 6 PCIEx4 ports [11] = Reserved	Defined with external 0_ohms resistor strap. R528 & R529
[15:14]	Reserved	R	--	Not implemented	
[18:16]	llk_RDRQOUT_MOD	RW	0x4	This 3-bit field defined the number of DOWN_stream Read_Request issued by the local PCI Express controller. This field is used for PCI Express tuning. (Bandwidth / latency)	FPGA EP control management. Used to control the down-stream Read_Request queue

<table>
<tr><th colspan="2">ILOC_PCIE_SW
IO_Bus : 0x014 - 0x017</th><th colspan="4">PCI Express Switch PEX8624</th></tr>
<tr><td></td><td></td><td></td><td></td><td><table>
<tr><td>000</td><td>No limitation Read Request</td></tr>
<tr><td>001</td><td>1 max outstanding Read_Request</td></tr>
<tr><td>010</td><td>2 max</td></tr>
<tr><td>011</td><td>4 max</td></tr>
<tr><td><b>100</b></td><td>8 max</td></tr>
<tr><td>101</td><td>16 max</td></tr>
<tr><td>110</td><td>32 max</td></tr>
<tr><td>Others</td><td>Reserved, invalid value</td></tr>
</table></td><td></td></tr>
<tr><td>[23:19]</td><td>Reserved</td><td>R</td><td>--</td><td>Not implemented</td><td></td></tr>
<tr><td>[24]</td><td>pex_GPIO_0</td><td>R</td><td>--</td><td>PEX8624 GPIO #12 (pin B15)</td><td rowspan="4">Hex Rotary switch status.</td></tr>
<tr><td>[25]</td><td>pex_GPIO_1</td><td>R</td><td>--</td><td>PEX8624 GPIO #13 (pin B17)</td></tr>
<tr><td>[26]</td><td>pex_GPIO_2</td><td>R</td><td>--</td><td>PEX8624 GPIO #14 pin B18)</td></tr>
<tr><td>[27]</td><td>pex_GPIO_3</td><td>R</td><td>--</td><td>PEX8624 GPIO #15 (pin R16)</td></tr>
<tr><td>[28]</td><td>pex_GPIO_4</td><td>R</td><td>--</td><td>PEX8624 GPIO #16 (pin V1)</td><td></td></tr>
<tr><td>[29]</td><td>pex_GPIO_5</td><td>R</td><td>--</td><td>PEX8624 GPIO #17 (pin T1)</td><td></td></tr>
<tr><td>[30]</td><td>pex_GPIO_6</td><td>R</td><td>--</td><td>PEX8624 GPIO #18 (pin U1)</td><td></td></tr>
<tr><td>[31]</td><td>pex_GPIO_7</td><td>R</td><td>--</td><td>PEX8624 GPIO #19 (pin P1)</td><td></td></tr>
</table>

Note The field llk_RDRQOUT_MOD is an important PCI Express related parameter. The remote Host READ performance will be directly related to this configuration mode control.

3.2.7 ILOC_SIGN Register

This register provides a 32-bit status information directly provided by the FPGA build date.

ILOC_SIGN IO_Bus : 0x018 - 0x01B		FPGA Signature			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	FPGA_Signature[31:0]	R	--	FPGA Signature . Consult IOxOS Technologies SA. FPGA_Signature[31:24] = Day FPGA_Signature[23.16] = Month FPGA_Signature[15:8] = Year FPGA_Signature[7:0] = Reserved(Sub version)	

Note This 32-bit field is assigned in [**tosca_glb_pkg.vhd**] before the XILINX ISE implementation process.

3.2.8 ILOC_GENCTL Register

This register provides general Control & Status informations.

ILOC_GENCTL IO_Bus : 0x01C - 0x01F	**PCIE General Control**				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[1:0]	fpga_BITSTR	R		FPGA Bit Stream loaded. Up to 4 different FPGA bit-stream can be loaded by the external CPLD. Bit-stream #0 is reserved as back-up stream.	Static Switch determine which one shall be loaded.
[3:2]	Reserved	R		Not used, reserved for extra bit-stream	
[4]	fpga_CONF_ERR	R		Initial FPGA Configuration process detected an error and FPGA is configured with the backup bit-stream.	
[5]	fpga_BIG	R		FPGA Configuration for big FF665 FPGA (SX50T & FX70T). Refer to chapter 2.5.1 and 2.5.2	Only positioned in hardware implementing large FPGA
[7:6]	Reserved	R		Not used	
[10:8]	ddr_STRAP[2:0]	R		On-board DDR2 installed Memory device [1xx] = x16 organisation (2 devices) [0xx] = x8 organisation (4 devices) [x00] = 256 Mbit [x01] = 512 Mbit [x10] = 1024 Mbit [x11] = 2048 Mbit	Default =001 DDR2 512 Mbit (SMEM = 256 MBytes)
[15:11]	Reserved	R		Not used	
[31:16]	FPGA Implementation	R		Fixed field determined at Synthesis & Place&Route process. Provide FPGA options selected	IOxOS Reserved

3.2.9 PCIE_MMUADD Register Address Pointer Register

This register provides the address pointer used to initialize the two INgress PCI Express MMU look-up tables. The look-up tables are composed of N page entries of 1o 4 MBytes. Each page entries owns a 36-bit page descriptor which shall be initialized with two sub-word of 18-bit.

- Non-Prefetchable INgress is up to 512* 1 MBytes pages (64-128-256-512) mapped with A32 BAR_2.
- Prefetchable INgress is up 1024* 4 MBytes pages (128-256-512-1024-2048-4096) mapped with A32/A64 BAR_0+(1)

The PCI_Express MEMORY window size allocated for the Non-Prefetchable and the Prefetchable are user selectable with static option SW501-(8:3).

PCIE_MMUADD IO_Bus : 0x020-0x023	**PCIE MMU Address Pointer**				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[0]	Reserved	R	0	Fixed to 00	LWORD aligned
[11:1]	pcie_MMUADD[10:0]	RW	0	11-bit Address pointer supporting the MMU Table	Auto incremented

PCIE_MMUADD IO_Bus : 0x020-0x023		**PCIE MMU Address Pointer**			
				initialization. This pointer is incremented after every access (Read or Write) through the PCIE_MMUDAT register.	address pointer.
[15:12]	Reserved	R	0	Not Implemented	
[16]	INgress SELECT	RW	0	INgress DPRAM Select 0 = Prefetchable INgress (512*D36) 1 = Non-Prefetchable INgress (1024*D36)	
[17]	Reserved	R	0	Not used	
[20:18]	No_PFMEM_Size	R	0	Non-Prefetchable Memory Size (BAR_2) 000 Not used 001 64 MBytes 010 128 MBytes 011 256 MBytes **100** 512 MBytes 101 Not used 110 Not used 111 Not used	
[21]	Reserved	R	0	Not used	
[22]	No_PFMEM_A64	R	0	Memory Non-Prefetchable is only 32-bit addressing.	
[23]	No_PFMEM_INgress	R	--	Memory Non-Prefetchable BAR_2 with PF_INgress MMU. = 1 : Active (Enabled) = 0 : Not implemented	
[25:24]	Reserved	R	0	Not used	
[28:26]	PFMEM_Size	R	--	Prefetchable Memory Size (BAR_0 + optionally BAR_1) 000 Not used 001 Not used 010 128 MBytes 011 256 MBytes **100** 512 MBytes 101 1024 MBytes 110 2048 MBytes 111 4096 MBytes	
[29]	Reserved	R	0	Not used	
[30]	PFMEM_A64	R	--	64-bit Addressing Memory Prefetchable	
[31]	PFMEM_INgress	R	--	Memory Prefetchable BAR_0 + (BAR1-1) with PF_INgress MMU. = 1 : Active (Enabled) = 0 : Not implemented	

3.2.10 PCIE_MMUDAT Register

This register provides the data access used to initialize the INgress PCI_Express MMU look-up table. The data access shall be handled with two 18-bit transaction.

PCIE_MMUDAT IO_Bus : 0x024-0x027		PCIE MMU Data Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[17:0]	pcie_MMUDAT[17:0]	RW	0	18-bit Data register, with previous register PCIE_MMUADD used to access the selected INgress MMU DPRAM	
[31:18]	Reserved	R	0	Not used	

The MMU Page Descriptor is composed of a single 36-bit word. The initialization is handled with dual 18-bit access through the PCIE_MMUDAT Register. Each n* MBytes page entry address its own DW Page Descriptor.

The following table sums up the Page Descriptor format :

Bit[]	Field	Description	Comments
		Hgh WORD Page Descriptor (pcie_MMUADD[0] = '1')	
[0]	Page_Enable	Enable the Page entry. If not set, the incoming transaction is rejected, even if the PCI Express BAR_1 address window is decoded.	Write Posting is not executed and Read_Request will not be completed.
[1]	Write_Enable	Enable Write access across this page entry. If not set, the Write Posting are rejected.	Allows to support write-protect area per page entry
[5:2]	Reserved	Not used	
[7:6]	Byte Swapping Policy	BYTE Swapping Policy. This 2-bit field specifies the Byte Swapping policy.(Little – Big Endian conversion) 00 No Swapping 01 LITTLE to BIG Endian Automatic Swapping BYTE : No Swapping WORD : D16 word swapping DW : BYTE & WORD Swapping 10 DW BYTE & WORD Swapping 11 DW swapping in QW access	Used only while targeting the VME64x Interface.1
[11:8]	Space_Type[3:0]	This 4-bit field encodes the Address Space targeted in destination. Refer to next table “Space_Type Address Encoding”	
[15:12]	SW_Destination	Destination Address Space Type. 4-bit field	Only two LSB bits used in the PEV_1100 implementation. Future implementation with larger central_SW with be encoded on 3 or 4-bit

Bit[]	Field	Description	Comments
		0000 PCI Express 0010 Shared Memory 0001 VME64X 0011 User Others Reserved, Invalid value	
[17:16]	Reserved	Not used (	
		Low WORD Page Descriptor (pcie_MMUADD[0] = '0')	
[19:18]	Reserved	Not used	
[35:20]	int_ADD[35:20]	Internal High Address remapping used to form a 36-bit Central_SW Address	Can be fixed to 0x0000'0000 while targeting Central_SW resources.

Table 3.2.: MMU page Descriptor

Note *Future implementation could extend the MMU Page Descriptor up to 64-bit (2 times 32-bit) to support 64-bit Central_SW Address and extra conditioning information.*

The following table sums-up the destination Address Space Type .

Space_Type	*VME64x Master*	*Shared Memory / USER*
0x0	Configuration Space	Memory standard
0x1	A16 : D08-D16-D32	Memory standard
0x2	A24 : D08-D16-D32	Reserved
0x3	A32: D08-D16-D32	Reserved
0x4	A32: BLT	Memory READ Cache entry #0
0x5	A32: MBLT	Memory READ Cache entry #1
0x6	A32: 2eVME	Memory
0x7	A32: 2eVME Fast	Reserved
0x8	A32: 2eSST_160	Reserved
0x9	A32: 2eSST_233	Reserved
0xa	A32: 2eSST_320	Reserved
0xb	A32: 2eSST_400 (Reserved)	
0xc	Reserved	Reserved
0xd	A24: Address Only	Reserved
0xe	A32: Address Only	Reserved
0xf	IACK Cycle	Reserved

Table 3.3.: Address Space Type Encoding

Note *In first release, advanced VME addressing mode are not supported. Provision is made for A64 addressing (A64:BLT, A64:MBLT and A64:2eSST) and for 2eSST Broadcast capabilities.*

3.2.11 ILOC_PCIe EP Status Register

This register provides control & status informations related to the XILINX embedded PCI Express EP. For complete description refer to XILINX technical documentation UG197 and UG350.

ILOC_PCIe_EP_STA IO_Bus : 0x028 - 0x02B		**PCI Express EP Control & Status**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[0]	PCIe_EP_Status	R	--	IO_SPACE_ENABLE	Refer to Xilinx UG197
[1]	PCIe_EP_Status	R	--	MEM_SPACE_ENABLE	
[2]	PCIe_EP_Status	R	--	BUS_MASTER_ENABLE	
[3]	PCIe_EP_Status	R	--	FIRST_CFG_WRITE_OCCURRED	
[4]	PCIe_EP_Status	R	--	RX_MAC_LINK_ERROR(0)	
[5]	PCIe_EP_Status	R	--	RX_MAC_LINK_ERROR(1)	
[6]	PCIe_EP_Status	R	--	MAC_LINK_UP	
[7]	PCIe_EP_Status	R	--	MAC_LINK_TRAINING	
[11:8]	PCIe_EP_Status	R	--	MAC_NEGOTIATED_LINK_WIDTH	
[12]	PCIe_EP_Status	R	--	DL_UP_DOWN(0)	
[13]	PCIe_EP_Status	R	--	DL_UP_DOWN(1)	
[15:14]	PCIe_EP_Status	R	--	Reserved	
[19:16]	PCIe_EP_Status	R	--	LTSSM_STATE	
[22:20]	PCIe_UPSEL	R	--	PCI Express UP_stream Port selection : 000 External Cabling Port P3 (Cable 100 MHz) 001 External Cabling Port P3 (Local 100 MHz) 010 XMC #1 011 XMC #2 100 Xilinx Virtex-5 FPGA 101 External Cabling Port P4 110 PEX_8112A in Reverse Mode. UP_stream on local PrPMC . 111 No UP_stream enabled. The PEX8624AA is disabled.	PEX_8624AA Static Selection. Defined by SW501-[8:6]
[23]	PCIe_Port1_NT	R	--	External Cabling Port P4 defined as Non-Transparent	PEX_8624AA Static Selection. SW501-5
[27:24]	Reserved	R	--	Not used	
[28]	TLI_FULLDUPLEX	RW	0	Xilinx PCI Express EP back-end interface = 0 : Full duplex disabled = 1 : Full duplex enabled (simultaneous Tx & Rx)	IOxOS debug support
[29]	8112AA_PG_RST	RW	1	PLX 8112AA EP (PCI Express to PCI Bridge) RESET Programmable. Set at Power-up , shall be clear by the PON_FSM microcode	
[30]	XILINX EP_PG_RST	RW	0	XILINX LX30T EP RESET Programmable. Set at Power-up , shall be clear by the PON_FSM	IOxOS debug support

ILOC_PCIe_EP_STA IO_Bus : 0x028 - 0x02B	PCI Express EP Control & Status				
				microcode. In case of PON_FSM disabled with SW502-2, it is kept inactive.	
[31]	GLOBAL EP_PG_RST	RW	0	GLOBAL RESET Programmable	IOxOS debug support

Note *The RESET control of the on-chip PCI Express is conditioned by the PON_FSM microcode execution. In normal operation mode, with PON_FSM not disabled, it is under the responsibility of the microcode to control the RESET sequencing.*

3.2.12 V5_SMON_ADDPT

This register provides address pointer to access the internal register of the System Monitor function embedded in the Virtex-5 FPGA. The Virtex-5 System Monitor integrates 128 16-bit registers access through the DRP "Dynamic Reconfiguration Port"

For complete description of internal resources refer directly to XILINX UG192 Virtex-5 FPGA System Monitor User's Guide.

V5_SMON_ADDPT IO_Bus : 0x040 - 0x043	Virtex_5 System Monitor Address Pointer				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[6:0]	V5SMON_ADDPT[6:0]	RW	0x00	System Monitor Register Address Pointer. Used with subsequent access Read/Write through next register.	
[30:7]	Reserved	R	--	Not Used	
[31]	V5SMON_AUTOINC	RW	0b0	While set, auto incremented V5SMON_ADDPT on every V5SMON_DATREG access is supported	

3.2.13 V5_SMON_DAT Register

This register provides the direct access to the Virtex-5 System Monitor resources. Refer to XILINX UG192 Virtex-5 FPGA System Monitor User's Guide.

V5_SMON_DAT IO_Bus : 0x044 - 0x047	Virtex_5 System Monitor Data Register				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[15:0]	V5SMON_DAT[15:0]	RW	0x00	Virtex-5 System Monitor Data Register. Used with previous address pointer.	
[31:16]	Reserved	R	--	Not Used	

3.2.14 V5_SMON_STA Register

This register provides status informations issued by the Virtex-5 System Monitor. Refer to XILINX

UG192 Virtex-5 FPGA System Monitor User's Guide.

V5_SMON_STA IO_Bus : 0x048 - 0x04B — **Virtex_5 System Monitor Status Register**

Bit[]	Function	R/W	Reset	Description	Comments
[2:0]	V5_SMON_ALM[2:0]	R	0b000	System Monitor Alarm Status	
[3]	V5_SMON_OT	R	0b0	System Monitor Over Temperature Status	
[8:0]	V5_SMON_Channel[4:0]	R	--	System Monitor ADC Channel Multiplexer	
[9]	V5_SMON_EOC	R	--	System Monitor End of Conversion	
[10]	V5_SMON_EOS	R	--	System Monitor End of Signal	
[11]	V5_SMON_BUSY	R	--	System Monitor ADB Busy	
[12]	V5_SMON_JTAGLOCK	R	--	System Monitor JTAG Port Locked	
[31:13]	Reserved	R	--	Not Used	

3.2.15 V5_PCIEP_ADDPT Register

This register provides address pointer to access the internal register of the PCI Express Endpoint function embedded in the Virtex-5 FPGA. The Virtex-5 PCI Express End point integrates 2048 32-bit registers, addressed with a 11-bit address bus. The internal resource access is handle through a dedicated “Management Interface”.

For complete description of internal resources refer directly to XILINX UG197 Virtex-5 FPGA PCI Express Integrated Endpoint User's Guide.

V5_PCIEP_ADDPT IO_Bus : 0x060 - 0x063 — **Virtex_5 PCI Express Endpoint Address Pointer**

Bit[]	Function	R/W	Reset	Description	Comments
[10:0]	V5PCIEP_ADDPT[10:0]	RW	0x00	Virtex-5 PCI Express Endpoint Address Pointer. Used with subsequent access Read/Write through next register.	
[30:11]	Reserved	R	--	Not Used	
[31]	V5PCIEP_AUTOINC	RW	0b0	While set, auto incremented V5PCIEP_ADDPT on every V5PCIEP_DATREG access is supported	

3.2.16 V5_PCIEP_DAT Register

This register provides the direct access to the Virtex-5 PCI Express Endpoint resources.

V5_PCIEP_DAT IO_Bus : 0x064 - 0x067 — **Virtex_5 PCI Express Endpoint Data Register**

Bit[]	Function	R/W	Reset	Description	Comments
[31:0]	V5SMON_DAT[15:0]	RW	0x00	Virtex-5 PCI Express Endpoint Data Register. Used with previous address pointer.	

3.2.17 V5_PCIEP_SEL Register

This register provides direct control to the Channel Credit selection of Virtex-5 PCI Express Endpoint resources. Refer to XILINX UG197 Virtex-5 FPGA PCI Express Integrated Endpoint User's Guide. 1680

V5_PCIEP_SEL IO_Bus : 0x068 - 0x06B	Virtex_5 PCI Express Endpoint Credit Select				
Bit[]	Function	R/W	Reset	Description	Comments
[6:0]	V5SMON_CRSEL[6:0]	RW	0x00	Virtex-5 PCI Express Endpoint Credit Selection Register.	
[31:7]	Reserved	R		Not Used.	

3.2.18 V5_PCIEP_RSLT Register

This register provides direct status information directly related to the Channel Credit Selection. Refer to XILINX UG197 Virtex-5 FPGA PCI Express Integrated Endpoint User's Guide.

V5_PCIEP_RSLT IO_Bus : 0x06C - 0x06F	Virtex_5 PCI Express Endpoint Credit Results				
Bit[]	Function	R/W	Reset	Description	Comments
[5:0]	MGMTPSO[5:0]	R		Not Used.	
[6]	MGMTPSO[6]	R		Transaction pending	
[7]	MGMTPSO[7]	R		Unsupported Request detected.	
[8]	MGMTPSO[8]	R		Fatal Error Detected	
[9]	MGMTPSO[9]	R		Non Fatal Error Detected	
[10]	MGMTPSO[10]	R		Correctable Error Detected	
[11]	MGMTPSO[11]	R		Detected parity Error (poisoned TLP)	
[12]	MGMTPSO[12]	R		Signalled System Error	
[13]	MGMTPSO[13]	R		Received Master Abort	
[14]	MGMTPSO[14]	R		Received Target Abort	
[15]	MGMTPSO[15]	R		Signalled Target Abort	
[16]	MGMTPSO[16]	R		Master Data parity Error	
[19:17]	Reserved	R		Not Used.	
[31:20]	V5SMON_CREDIT[11:0]	RW	0x000	Virtex-5 PCI Express Endpoint Updated Credit information.	

1685

3.2.19 PMC_XMC CSR Register

This register provides control & status information related to the IEEE 1386.1 PMC and XMC VITA42 mezzanine

PMC_XMC IO_Bus : 0x0C0 - 0x0C3		**PMC & XMC Control & Status**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[0]	pmc_INTA_1	R	0b0	PMC#1 INTA Status	
[1]	pmc_INTA_2	R	0b0	PMC#2 INTA Status	
[7:2]	Reserved	R		Not Used.	
[9:8]	pmc_PCICLKSEL[1:0]	RW	0b00	PCI Clock Frequency selection. [00] = 33 MHz [01] = 40 MHz (Not supported) [10] = 55 MHz (Not supported) [11] = 66 MHz (Not supported)	Fixed to “00”
[10]	plx_M66EN	R	--	Local PCI 66 MHz Enable. Status only, value derived from pmc_PCICLKSEL[1:0]	
[11]	plx_FORWARD	R	0b0	PEX8112 Bridge Control. Status information only. The control of the signal is defined by the SW-500	
[12]	pmc_PCIRESET	R	0b0	Local PCI RESET. Status information only.	
[13]	plx_CWAKEINn	RW	0b0	PEX8112 Bridge, direct control signal	
[14]	plx_BAR0ENB	RW	0b0	PEX8112 Bridge, direct control signal	
[15]	plx_PERST	R	0b1	PEX8112 Bridge. Status information only.	
[16]	xmc_1_PRESENT	R	--	VITA-42 XMC Status Information	
[17]	xmc_1_RSTIN	RW	0b1	VITA-42 XMC RESET Command	
[18]	xmc_1_ROOT	RW	0b0	VITA-42.3 XMC ROOT	
[23:19]	Reserved	R		Not Used.	
[24]	xmc_2_PRESENT	R	--	VITA-42 XMC Status Information	
[25]	xmc_2_RSTIN	RW	0b1	VITA-42 XMC RESET Command	
[26]	xmc_2_ROOT	RW	0b0	VITA-42.3 XMC ROOT	
[31:27]	Reserved	R	-	Not Used.	

Note *PMC Interrupt are at this point only monitored. Local PCI Interrupt generation could be easily implemented on future release.*

3.2.20 I2C CTL_X Registers

These following control & status registers provides interface support for the five on-board I2C Serial Bus listed in following items :

a) I2C PLX PEX8624 PCI Express Initialization Bus. Need 4 Bytes Command and Data.

b) Two I2C Buses for on-board Temperature Monitor LM86. Need single Byte Command and Data.

c) Two XMC IPMI SMB_Bus. (One per XMC site)

The I2C is implemented through a register based interface with limited functionality to support PEV_1100 on-board device.

- Address + Command

- Address + K*Command + N*Data_Write (N = 1 to 4, K = 1 to 4)
- Address + N*Data_Read (N = 1 to 4)

The Control register provides all control and status related to the I2C cycle generation.

I2C_CTL IO_Bus : 0x0100 - 0x0103		**I2C Control**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[6:0]	I2C_ADD[6:0]	RW	0x00	I2C 7-bit Address field	
[7]	Reserved	R		Not Used	
[10:8]	I2C_ADD[9:7]	R	0x0	I2C 10-bit Address extension field	
[15:11]	Reserved	R		Not Used	
[17:16]	I2C_CMDSIZ[1:0]	RW	0b00	I2C Byte Count for CMD field (1 to 4)	
[19:18]	I2C_DATSIZ[1:0]	RW	0b00	I2C Byte Count for DATW field (1 to 4)	
[21:20]	I2C_EXECSTA[1:0]	R	0b00	I2C Interface Cycle Execution Status 00 : I2C Idle 01 : I2C Cycle Running 10 : I2C Cycle Executed without Error 11 : I2C Cycle Executed with Error (Aborted)	
[23:22]	I2C_TRIG[1:0]	W	0b00	I2C Bus Trigger Command. 01 : I2C Cycle ADD + n* CMD 10 : I2C Cycle ADD + n* CMD +n*DATW 11 : I2C Cycle ADD + n* DATR	
[25:24]	I2C_SPEED[1:0]	RW	0b00	I2C Bus Speed Selection 00 : Standard-mode **Sm** (100 Kbit/s) 01 : Fast-mode **Fm** (400 Kbit/s) 10 : Fast-mode Plus **Fm+** (1 Mbit/s) 11 : High-speed mode Plus **Hs** (3.4 Mbit/s)	Only mode Sm and Fm need to be supported.
[26]	SMB_SPEED	RW	0b0	SMB Bus Speed Selection 0 : Slow-mode **Sm** (10 kbit/s) 1 : Fast-mode **Fm** (100 kbit/s)	
[27]	THERM_CRITn	R	0b0	LM86 SMB Bus T_CRIT active low output (both temperature sensors)	
[28]	THERM_ALERTn	R	0b0	LM86 SMB Bus interrupt active low output (both temperature sensors)	
[31:29]	I2C_Port_SEL[2;1]	RW	0x0	On-board I2C Port Selection. 000 : PEX8624 PCI Express Switch 010 : LM86 Temperature Monitor #1 011 : LM86 Temperature Monitor #2 100 : IPMI XMC#1 101 : IPMI XMC#2	Providion for three additional port

The Command register holds up to four (4) Command Bytes, whose be used in the subsequent I2C transaction. The number of Command Bytes is defined in the field I2C_CMDSIZ[1:0] of the I2C_CTL register.

I2C_CMD IO_Bus : 0x0104 - 0x0107		**I2C Command**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[7:0]	I2CCMD_BYTE_0	RW	0x00	I2C Bus Command Byte #0	1st Byte

I2C_CMD IO_Bus : 0x0104 - 0x0107		I2C Command			
[15:8]	I2CCMD_BYTE_1	RW	0x00	I2C Bus Command Byte #1	2nd Byte
[23;16]	I2CCMD_BYTE_2	RW	0x00	I2C Bus Command Byte #2	3rd Byte
[31:24]	I2CCMD_BYTE_3	RW	0x00	I2C Bus Command Byte #3	4th Byte

The Data Write register holds up to four (4) Data Bytes, whose be used in the subsequent I2C Write transaction. The number of Data Bytes is defined in the field I2C_DATSIZ[1:0] of the I2C_CTL register.

I2C_DATW IO_Bus : 0x0108 - 0x010B		I2C Data Write			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[7:0]	I2C_DATW_BYTE_0	RW	0x00	I2C Bus Data Write Byte #0	1st Byte
[15:8]	I2C_DATW_BYTE_1	RW	0x00	I2C Bus Data Write Byte #1	2nd Byte
[23;16]	I2C_DATW_BYTE_2	RW	0x00	I2C Bus Data Write Byte #2	3rd Byte
[31:24]	I2C_DATW_BYTE_3	RW	0x00	I2C Bus Data Write Byte #3	4th Byte

The Data Read register stores four (4) Data Bytes received from subsequent I2C Read transaction.

I2C_DATR IO_Bus : 0x0110 - 0x013B		I2C Data Read			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[7:0]	I2C_DATR_BYTE_0	R	0x00	I2C Bus Data Read Byte #0	1st Byte
[15:8]	I2C_DATR_BYTE_1	R	0x00	I2C Bus Data Read Byte #1	2nd Byte
[23;16]	I2C_DATR_BYTE_2	R	0x00	I2C Bus Data Read Byte #2	3rd Byte
[31:24]	I2C_DATR_BYTE_3	R	0x00	I2C Bus Data Read Byte #3	4th Byte

3.2.21 Private LOCAL_ITC Controller

A dedicated Interrupt controller ITC is embedded in the LOCAL section. Only 6 Interrupt Source are connected to the ITC.

Following table sums up the ILOC (PCIe_EP) Block Interrupt source allocation

ITC INT#	Mode	Assignation	Comments
#0	LEVEL	V5_SMON_ALM[0] System Monitor Alarm	XILINX Virtex-5 System Monitor
#1	LEVEL	V5_SMON_ALM[1] System Monitor Alarm	
#2	LEVEL	V5_SMON_ALM[2] System Monitor Alarm	
#3	LEVEL	V5_SMON_OT System Monitor Over Temp	XILINX Virtex-5 System Monitor
#4	LEVEL	LM86 CRIT	On board LM86 Thermal Monitor.
#5	LEVEL	LM86 THERM Alarm	
#6	LEVEL	Not used	Ready for instantiation at top level
#7	LEVEL	Not used	
#8	LEVEL	Not used	
#9	LEVEL	Not used	
#10	LEVEL	Not used	
#11	LEVEL	Not used	
#12	LEVEL	Not used	
#13	LEVEL	Not used	
#14	EDGE	Not used	
#15	EDGE	Not used	

Table 3.4. : LOCAL (PCIe EP) Block ITC Interrupt Sources

3.2.22 Local Signature_ROM

This area provides a non-volatile information related to the LOCAL's block. Refer to chapter X.X for format description.

- Device_ID and Revision_ID.
- Any valuable TBD implementation informations.

Signature_ROM IO_Bus : 0x380 - 0x3FF		Standard ROM area			
Location	Function	R/W	Reset	Description	Comments
0x0:[7:0]		R	--	Implementation Revision	
0x1F:[31:24]	Not used	R	--	Not used ...	

3.3 VME64X CSR Resources

The VM64X CSR Resources are mapped through a dual-port interface as :

- IO_Bus Device#2 (0x400-0x7FF) 1 KBytes
- VME64X CR-CSR Space (512 KBytes), offset determined by VME64x Geographic address (GA[4:0] & GAP) or with static option SW500-[8:4] if the VME64X mode is disabled.

The VME64X Private registers provides the facilities to configure the VME64X Master/Slave Interface. The following tables sums up the VME64X Private Register Mapping.

Offset IO_Bus	Size	Register_Name	Description	Comments
0x400 - 0x403	0x4	**PVME_SLOT1**	VME64X Slot-1 related functions as Arbiter, BTO and Static option switches status.	
0x404 - 0x407	0x4	**PVME_MASCSR**	VME64X Master Port related control and status	
0x408 - 0x40B	0x4	**PVME_SLVCSR**	VME64X Slave Port related control and status	
0x40C - 0x40F	0x4	**PVME_INTG**	VME64X Interrupt Generator	
0x410 - 0x413	0x4	**PVME_MMUADD**	VME64x INgress MMU Address pointer.	
0x414 - 0x417	0x4	**PVME_MMUDAT**	VME64x INgress MMU Data Register	
0x418 - 0x41B	0x4	**PVME_ADDERR**	VME64x Address Error Register	These two registers provide common status info in case of VME Error detected
0x41C - 0x41F	0x4	**PVME_ADDERR**	VME64x Status Error Register	
0x420 - 0x423	0x4	**PVME_LOCMON**	VME64x Location Monitor Address	
0x424 - 0x427	0x4	**PVME_LOCK**	VME64x ADOH/LOCK support	VME64x ADOH & LOCK support
0x428 - 0x42B	0x4	**PVME_RMW_MODE**	VME64x RMW Operation control	VME RMW Compare and Swap (CAS)
0x42C - 0x42F	0x4	**PVME_RMW_ADD**	VME64x RMW Address Pointer	
0x430 - 0x433	0x4	**PVME_RMW_DATCMP**	VME64x RMW Data Compare	
0x434 - 0x437	0x4	**PVME_RMW_DATUPT**	VME64x RMW Data Update	
0x440 - 0x443	0x4	**PVME_GLTIM_CSR**	Global Timer Control & Status Register	
0x444 - 0x447	0x4	**PVME_GLTIM_DBG**	Global Timer Debugging support	
0x448 - 0x44B	0x4	**PVME_GLTIM_CNT2**	Global Timer Pre-scaler 17-bit	
0x44C - 0x44F	0x4	**PVME_GLTIM_CNT1**	Global Timer Main 32-bit counter (1 [ms]	
0x480 - 0x48F	0x10	**PVME_ITC**	VME64x ITC Interrupt Pending	Refer to ITC chapter
0x490 - 0x55F		**Reserved**	Reserved for additional facilities	
Following registers are also mapped on VME64X CR/CSR Space				
0x560	0x1	**VMECSR_ADER0_3**	Address Space Relocation Register VME A32 MSB	VME64x A32 Base Address Definition. Refer to VME64x Specification.

Offset IO_Bus	Size	Register_Name	Description	Comments
0x564	0x1	**VMECSR_ADER0_2**	Address Space Relocation Register VME A32	
0x568	0x1	**VMECSR_ADER0_1**	Address Space Relocation Register VME A32	
0x56C	0x1	**VMECSR_ADER0_0**	Address Space Relocation Register VME A32 LSB	
0x570 - 0x5F0		**Reserved**		PEV_1100 does not incorporate any CRAM resources.
0x5F4	0x1	**VMECSR_BCR**	VME64x CR/CSR Bit Clear Register	VME64x main Control & Status Control
0x5F8	0x1	**VMECSR_BSR**	VME64x CR/CSR Bit Set Register	
0x5FC	0x1	**VMECSR_BAR**	VME64x CR/CSR Base Address Register	
0x780 - 0x7FF	0x80	**Signature_ROM**	Defined ROM area for software signature	

The VME64X CR/CSR resources are accessible from two different path (IO_Bus Slave and VME64X Slave) The resource mapping is slightly different when addressed from the internal IO_Bus or from the VME64X CR/CSR Slave interface.

The VME64X CR/CSR is defined by the ANSI/VITA 1.0-1994 [2002] and complemented with ANSI/VITA 1.1-1997 [2003] specification.

The following tables sums up the VME64X CR/CSR Register Mapping (512 KBytes area). The register mapping is described in big-endian (VME64X natural endian)

Offset CR/CSR	Size	Register_Name	Description	Comments
0x0'0000 - 0x0'0FFF	0x1000	**VME_CR**	VME64 0x80 Bytes, PROM Signature	Fixed read only ROM space. Refer to ANSI/VITA 1-1994(R2000) Table 2-32
0x01000 - 0x7'0FFF	....	**Reserved**		
0x7'1000 - 7'1FFF		**IO_Bus Bridge**	4 KBytes directly mapped to the internal IO_Bus infrastructure. Base + 0x000-0x3FF = Local TCSR 0x400-0x7FF = VME64x TCSR 0x800-0xBFF = SMEM-IDMA TCSR 0xC00-0xFFF = USER TCSR	IO_Bus arbitrated access. Provide full IO_Bus mapping over the VME_Bus
0x7'2000 - 7'2FFF		**IO_Bus Extension**	Reserved for IO_Bus addressing range area	
0x7'F000 - 7'F3FF		**Direct VME CTL**		
0x7'FF63	0x1	**VMECSR_ADER0_3**	Function_0 Address Mapping Registers	Only VME A32 Slave window is supported.
0x7'FF67	0x1	**VMECSR_ADER0_2**	Idem	
0x7'FF6B	0x1	**VMECSR_ADER0_1**	Idem	
0x7'FF6F	0x1	**VMECSR_ADER0_0**	Idem	
0x7'FF70 - 0x7'FFF6		**Reserved**	Not implemented	The CRAM resources is not supported.

Offset CR/CSR	Size	Register_Name	Description	Comments
0x7'FFF7	0x1	**VMECSR_BCR**	VME BIT Clear Register	
0x7'FFFB	0x1	**VMECSR_BSR**	VME BIT Set Register	
0x7'FFFF	0x1	**VMECSR_BAR**	VME Geographic Base Address Register. Read only register reflecting the VME64X GA[4:0] pins.	

3.3.1 Private VME Slot_1 Register

This register provides control and status related to the VME64X slot_1 function.

PVME_SLOT1 IO_Bus : 0x400-0x403 CR/CSR : 0x7F003-0x7F000		**Slot_1 & General Control & Status**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[1:0]	vmeSLOT1_CARB[1:0]	RW	00	VME Central Arbiter Mode of operation : 00 Not pipelined PRI Mode. 01 Not pipelined RRS Mode. 10 Pipelined PRI Mode. 11 Pipelined RRS Mode. When "Pipelined" mode is enabled, the VME Arbiter issued VME BGn[x] while the current VME cycle is still active. This mode of operation can provide better Bus utilisation is some situation. ***Note*** *Some VME Master does not support the arbitration pipeline.*	
[2]	vmeARBTO_FL	RW	0	VME central Arbiter time-out flag. Status set while a BGOUT[x] was issued without Bus Mastership confirmation. (Arbitration time-out fixed to 4[us]). Flag is clear by writing '1' over the it. This condition should never occurs. If detected , a VME BG daisy chain problem can be present.	
[3]	vmeSLOT1_Enable	R	0	VME Slot_1 Enabled status flag. The VME Slot_1 control is determined by the VME Geographical address or with external SW500-2 switch option.	
[4]	vmeRTO_MOD	RW	0	VME Arbitration Requester time-out mode. A dedicated time-out logic monitors the VME Arbitration Request phase. This time-out logic can be disabled by setting vmeRTO_MOD. 0 512 [us] 1 Disable. No time-out protection ***Note*** *The time-out value shall cover standard system implementation. VME arbitration mastership latency over 512[us] is abnormal.*	
[5]	vmeRTO_FL	RW	0	VME Arbitration Request time-out flag. Status set	

PVME_SLOT1 IO_Bus : 0x400-0x403 CR/CSR : 0x7F003-0x7F000	Slot_1 & General Control & Status				
				while a RTO time out occurs. Flag is clear by writing '1' over the it.	
[7:6]	Reserved	R	0	Not implemented, Read at 0b0000	
[9:8]	vmeBTOMOD[1:0]	RW	00	VME BTO /2eBTO Slot-1 value : 00 – 16 [us] 01 – 64 [us] 10 – 128 [us] 11 – 256[us] ***Note*** *This time-out covers only the VME cycle phase. A dedicated time-out is assigned to the VME Arbitration Request phase*	Cassidian specific
[10]	vmeCYBTO_FL	RWc	0	VME BTO/2eBTO time-out flag. Status set while a BTO/2eBTO time out. Flag is clear by writing '1' over the it.	
[14:11]	Reserved	R	0	Not implemented, Read at 0b0000	
[15]	vmePEVRST_Ena	RW	0	This control bit enable VME RESET assertion while a PCIe RESET is received. 0 – VME64x SYSRESET# Disabled 1 – VME64x SYSRESET# Enabled	Cassidian specific
[20:16]	vme64x_GEO[4:0]	R	0	Status field VME64X Geographic Address or Static switch field while VME64X is not enabled.	
[23:21]	Reserved	R	0	Not implemented, Read at 0b000	
[24]	static_SW_64X	R	-	Static switch Enable VME64X Mode	
[25]	static_SW_SLOT1	R	-	Static switch enabling VME Slot_1 Function	
[26]	static_SW_SYSRST	R	-	SW500-1 status , enabling VME SYSRESET# Generation from on-board logic.	
[27]	vme64x_Auto_ID	R	-	VME64X Auto_ID Mode enabled. Set while “static_SW_64X = OFF” and	
[29:28]	vme64x_Auto_STA	R	00	VME64X Auto_ID FSM status 00 – Idle or Auto_ID mode disabled 01 – VME IRQ2# pending. Waiting for IACK 10 – Waiting for CRCSR Base Register update (write) from VME Slave 11 – Auto_ID completed. CRCSR Base Register updated with new programmed value	
[30]	vme64x_RxSYSFAIL	R	-	VME64X SYSFAIL status	

PVME_SLOT1 IO_Bus : 0x400-0x403 CR/CSR : 0x7F003-0x7F000				**Slot_1 & General Control & Status**	
				0 VME64x SYSFAIL not asserted 1 VME64x SYSFAIL asserted	
[31]	vme64x_Auto_ID_GO	W	0	Command to start Auto_ID Semi-automatic process → IRQ2# assertion + SYSFAIL# deassertion	
	vme64x_Auto_ID_Semi	R	0	VME64X Auto_ID Semi-Automatic Mode	

3.3.2 Private VME_Master Register

This register provides control and status associated with the VME64X Master function.

PVME_MASCSR IO_Bus : 0x404-0x407 CR/CSR : 0x7F007-0x7F004				**VME Master CSR Register**	
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[1:0]	vmas_ARBMOD[1:0]	RW	00	VME Arbitration Requester Mode. This field can be changed dynamically, allowing to control the “No_Release ” operation. 00 **RWD** Release When Done. 01 **ROR** Release On Request. 10 **FAIR** & Release When Done 11 **No Release**.	Complete VME64X Arbitration Requester
[3:2]	vmas_ARBLEV[1:0]	RW	00	VME Arbitration Requester Level. 00 BREQ Level 0 01 BREQ Level 1 10 BREQ Level 2 11 BREQ Level 3 All VME Master transactions (READ, WRITE and IACK) use the same BREQ Level. (No possible differentiation)	
[4]	vmas_BACKOFF	RW	0	While set, any VME cycle ended with a RETRY* acknowledge will force a new VME Arbitration sequence.(Forced Release) ***Note*** *This mode of operation shall be set to recover some dead-lock situation in addressed VME Slave.*	

<table>
<tr><th colspan="2">PVME_MASCSR
IO_Bus : 0x404-0x407
CR/CSR : 0x7F007-0x7F004</th><th colspan="4">VME Master CSR Register</th></tr>
<tr><td>[5]</td><td>vmas_SUPER</td><td>RW</td><td>0</td><td>While set, use the VME AM code assigned by Supervisory Mode. (0x2F, 0x2D, 0x3D, 0xD, 0xC). When clear, use User VME AM code.
<b>Note</b> <i>Temporary, information will be assigned in the MMU scatter-gather page descriptor.</i></td><td></td></tr>
<tr><td>[6]</td><td>vmas_MAXRETRY</td><td>RW</td><td>0</td><td>While set, the VME RETRY on same back-end transaction is limited to a pgm value (32, 64, 128, 256) defined by "vmas_MAXRTCNT[1:0]" control field. Transaction is than terminated with an Error condition.(Idem as BERR)
While clear, the VME RETRY on same back-end transaction is limited by a time based value defined with "vmas_MRTY_TIM[7:0"] control field.</td><td>Cassidian specific Updated to support selectable maximum RETRY</td></tr>
<tr><td>[7]</td><td>vmas_READERR_BP</td><td>RW</td><td>0</td><td>While set, VME BERR Read are terminated normally status over the PCI_Express EP. Allows to disable the "Machine Check Exception"on the Host controller.
<b>Note</b> <i>PCI Express chips-set usually support READ Error without generating catastrophic "Machine Check Exception"</i></td><td></td></tr>
<tr><td>[8]</td><td>vmas_REJ_FL</td><td>RW</td><td>0</td><td>VME pending cycle rejected flag, due to vmas_ENABLE = '0'.
Flag is clear by writing '1' over the it.</td><td></td></tr>
<tr><td>[9]</td><td>vmas_HW16_SW</td><td>RW</td><td>0</td><td>MVME Master D16 Access special swapping mode. Used only while the Swapping_Mode defined by the INgress MMU is selected to "10"
While set, D16 access is only BYTE swapped.</td><td>This special operation mode is provided for legacy VME D16 only compatibility.</td></tr>
<tr><td>[10]</td><td>vmas_IACK_DW</td><td>RW</td><td>0</td><td>VME IACK Master (programmed_IO) enabled for DW Status_ID
= 0 : Local ADD[3:1] → VME IACK A[3:1]
= 1 : Local ADD[4:2] → VME IACK A[3:1]</td><td></td></tr>
<tr><td>[11]</td><td>Reserved</td><td>R</td><td>0</td><td>Not Implemented</td><td></td></tr>
<tr><td>[13.12]</td><td>vmas_READ_SMP[1:0]</td><td>RW</td><td>0</td><td>This 2-bit field allows to select the VME Data Read sampling point, related to the received VME DTACK# edge
<table>
<tr><th>READ_SMP</th><th>Sampling point (timing)</th></tr>
<tr><td>00</td><td>+ 12[ns] (default)</td></tr>
<tr><td>01</td><td>+ 18[ns]</td></tr>
<tr><td>10</td><td>+ 24[ns]</td></tr>
<tr><td>11</td><td>+ 0[ns]</td></tr>
</table></td><td>New Cassidian</td></tr>
<tr><td>[15:14]</td><td>vmas_MAXRTCNT[1:0]</td><td>RW</td><td>00</td><td>VME Maximum RETRY accepted. Used with control bit "vmas_MAXRETRY"
<table>
<tr><th>MAXRTCNT</th><th>Number of maximum RETRY</th></tr>
<tr><td>00</td><td>256 Retries</td></tr>
</table></td><td>Cassidian specific</td></tr>
</table>

<table>
<tr><td colspan="2"><b>PVME_MASCSR</b>
IO_Bus : 0x404-0x407
CR/CSR : 0x7F007-0x7F004</td><td colspan="4"><b>VME Master CSR Register</b></td></tr>
<tr><td></td><td></td><td></td><td></td><td><table><tr><td>01</td><td>128 Retries</td></tr><tr><td>10</td><td>64 Retries</td></tr><tr><td>11</td><td>32 Retries</td></tr></table></td><td></td></tr>
<tr><td>[17:16]</td><td>vmas_CMPLCRS[1:0]</td><td>RW</td><td>00</td><td>Completion TLP mode of operation for VME Master READ transaction with RETRY#.<table><tr><th>CMPL_CRS</th><th>Mode of operation</th></tr><tr><td>00</td><td>No Completion</td></tr><tr><td>01</td><td>Completion with Compl. Status = CRS on maximum VME RETRY (256) VME RETRY# acknowledge.</td></tr><tr><td>10</td><td>Completion with Compl. Status = CRS on 1st VME RETRY# acknowledge.</td></tr><tr><td>11</td><td>Completion with Compl. Status = CRS on fourth VME RETRY# acknowledge.</td></tr></table></td><td>Cassidian specific</td></tr>
<tr><td>[19:17]</td><td>Reserved</td><td>R</td><td>0</td><td>Not Implemented</td><td></td></tr>
<tr><td>[27:20]</td><td>vmas_MRTY_TIM[7:0]</td><td>RW</td><td>0x00</td><td>This control field defines a time value when the VME transaction is retried automatically. Used with control bit “vmas_MAXRETRY = '0”<table><tr><th>MRTY_TIM</th><th>Number of maximum RETRY</th></tr><tr><td>0x00</td><td>Infinite time</td></tr><tr><td>0x01</td><td>= 9 [us]</td></tr><tr><td>0x02</td><td>= 17 [us]</td></tr><tr><td>...</td><td>= (N* 8) + 1) [us]</td></tr><tr><td>0xFF</td><td>= 2041 [us]</td></tr></table></td><td>Cassidian specific</td></tr>
<tr><td>[30:28]</td><td>Reserved</td><td>R</td><td>0</td><td>Not Implemented</td><td></td></tr>
<tr><td>[31]</td><td>vmas_ENABLE</td><td>RW</td><td>1</td><td>VME64X VME Master Enable</td><td>Global VME Enable</td></tr>
</table>

3.3.3 Private VME_Slave Register

This register provides control and status associated with the VME64X Slave function.

PVME_SLVCSR
IO_Bus : 0x408-0x40B
CR/CSR : 0x7F00B-0x7F008

VME Slave CSR Register

Bit[]	Function	R/W	Reset	Description	Comments
[3:0]	vslv_WSIZ[2:0]	RW	0	VME A32 Slave window size. This 3-bit field determines the VME A32 Slave window occupied by the unit. 0000 16 MBytes as 16 Pages of 1 MBytes 0001 32 MBytes .... 0010 64 MBytes .... 0011 128 MBytes .... 0100 256 MBytes ... 0101 512 MBytes as 512 Pages of 1 MBytes 0110 1024 MBytes as 512 Pages of 2 MBytes. 0111 2048 MBytes as 512 Pages of 4 MBytes. 1000 1 MBytes as 1 Page of 1 Mbytes 1001 2 MBytes .... 1010 4 MBytes .... 1011 8 MBytes .... 1100 16 MBytes .... 1101 32 MBytes .... 1110 64 MBytes .... 1111 128 MBytes .... The VME Slave A32 window is divided in pages of 1, 2 or 4 MBytes, conditioned by vslv_WSIZ[3:0] While vslv_WSIZ[3] = '0' the VME A32 base address shall be aligned on window size boundary. While vslv_WSIZ[3] = '1' the VME A32 base address shall be aligned on 1 MBytes boundary.	
[4]	vslv_RETRY_Legacy	RW	0	VME Slave RETRY Acknowledge. While set, VME RETRY is never used for SLT and BLT data transfers. Shall be set while interaction with 1st generation of VME board.	
[5]	Reserved	R	0	Not Implemented	
[7:6]	WP_Max_BURST	RW	00	Write_Posting burst size. This two bit field allows to limit the maximum Central_SW write-posting packet size. This control field in not used while targeting the PCI Express EP. (PCI Express has its own burst size limitation mechanism)	

PVME_SLVCSR IO_Bus : 0x408-0x40B CR/CSR : 0x7F00B-0x7F008		**VME Slave CSR Register**			
				00 Max 1 KBytes 01 512 Bytes 10 256 Bytes 11 128 Bytes **Note** *Only used for system latency tuning.*	
[8]	RRSP_Invalid	RWc	0	Read Response Error flag status. Incoherent/Invalid Response	Status flag cleared while '1' written over
[9]	RRSP_Unexpected	RWc	0	Read Response Error flag status. Unexpected Response	Status flag cleared while '1' written over
[10]	RRSP_Wrong_ACK	RWc	0	Read Response Error flag status. Wrong Acknowledge	Status flag cleared while '1' written over
[11]	RRSP_ Out_of_Date	RWc	0	Read Response Error flag status. Out_of_Date Response	Status flag cleared while '1' written over
[12]	INTHVME_Enable	RW	0	Enable on-board Interrupt Handler across VME Slave Interface . While set ('1') the PCI Express MSI IACK can be handled through the VME Slave Bridge.	
[13]	vslv_RDRS_TOFL	RWc	0	Read Response time-out flag. Set in case of Read RESPONSE time-out occurred. Flag is clear by writing '1' over it.	Cassidian specific
[15:14]	vslv_RDRS_TO	RW	00	This two bit field define the Read_response time-out. A Read Response shall be received within this time value 00 256[us] 01 512[us] 10 1024 [us] 11 2048 [us] **Note** *Only used with vslv_READ_SPMOD = '1'*	Cassidian specific
[23:16]	vslv_RDRQ_TO[7:0]	RW	0x00	This eight bit field define in microsecond, when the current VME Read transaction is suspended by issuing VME RETRY# **Note** *Only used with vslv_READ_SPMOD = '1'*	Cassidian specific
[24]	vslv_READ_SPMOD	RW	0	VME Read SUSPEND Mode. While set, the current VME READ transaction is suspended with retry after the Read_Request Time-out (defined with control field vslv_RDRQ_TO[7:0]) has elapsed. While clear, the current VME READ transaction is not suspended.	Cassidian specific
[30:25]	Reserved	R	0	Not Implemented	
[31]	vslv_ENABLE	RW	0	VME64X VME Slave Enable.	Global VME Enable

- Status flags RRSP_Unexpected, RRSP_Wrong_ACK, RRSP_ Out_of_Date are directly derived from the PCI_Express implementation.

3.3.4 Private VME_INTG Register

This register provides the control & status related to the VME Interruption control.

PVME_INTG IO_Bus : 0x40C-0x40F CR/CSR : 0x7F00F-0x7F00C		**VME Interrupt Generator Register**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[7:0]	vme_INTG_STATUS	RW	0x00	VME Interrupt Generator Status ID LSB. This 8-bit field is supplied during the related IACK cycle, acknowledging the Interrupt.	
[10:8]	vme_INTG_LEVEL	RW	0x0	VME Interrupt Generator Level, 3-bit encoded value	Value “000” Not used
[11]	vme_INTG_IP	R	0	VME Interrupt Pending status flag	Default = “000”
[15:12]	vme_INTG_CMD	RW	0x0	VME Interrupt Generator Command [0x0] : No Action [0x1] : Set VME Interrupt [0x2] : Clear VME Interrupt ... [0x9] : Pgm/Init Status-ID[15:8] [0xA] : Pgm/Init Status-ID[23:16] [0xB] : Pgm/Init Status-ID[31:24] .... Others : Undefined	
[19:16]	vme_INTG_FIFOCNT	R	0x0	INTG FIFO List word counter. Reserved for future improvement	
[22:20]	Reserved	R	0	Not implemented	
[23]	vme_INTG_MOD	RW	0	VME64 Interrupt Generation Mode [0] Direct Register Programmed [1] FIFO List Enable. (Not supported) ***Note*** *Provision is made to support a front end FIFO allowing to post up to 16 VME Interrupt.*	
[24]	Reserved	R	0	Not implemented	
[31:25]	vme_INTPD[7:1]	R	0x00	VME Interrupt pending. Direct status information from the VME64x backplane.	

3.3.5 Private VME_MMU Address Pointer Register

This register provides the address pointer used to initialize the INgress VME MMU look-up table. The INgress VME MMU is implemented with a dual-port SRAM (512 * D64) , supporting up to 512 page entries. The page size is dependant of the VME A32 Window size selected.

PVME_MMUADD IO_Bus : 0x410-0x413 CR/CSR : 0x7F013-0x7F010	**VME MMU Address Pointer**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[0]	Reserved	R	0	Fixed to 0	WORD aligned
[11:1]	vme_MMUADD[9:0]	RW	0	10-bit Address pointer supporting the MMU Table initialization. This pointer is incremented after every access (Read or Write) through the PVME_MMUDAT register. MMUADD[1:0] = 00 : Page_DESC[15:0] 01 : Page_DESC[31:16] 10 : Page_DESC[47:32] 11 : Page_DESC[63:48] ***Note*** *The complete Page Descriptor initialization requires 4 consecutive loads*	Auto incremented address pointer.
[31:12]	Reserved	R	0	Not Implemented	

3.3.6 Private VME_MMU Data Register

This register provides the address pointer used to initialize the Ingress VME MMU look-up table.

PVME_MMUDAT IO_Bus : 0x414-0x417 CR/CSR : 0x7F017-0x7F014	**VME MMU Data Register**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[15:0]	vme_MMUDAT[15:0]	RW	0	16-bit Data register, associated with previous register PVME_MMUADD used to access the MMU DPRAM	
[31:16]	Reserved	R	0	Fixed to 00	

1775 The MMU Page Descriptor is composed of two consecutive DW, forming a 64-bit QW. The initialization is handled with two access through the PVME_MMUDAT Register. Each 4 MBytes page entry address owns its QW Page Descriptor. (DPRAM : 512*D64 - 2048*D16)

The following table sums up the Page Descriptor format :

Page Descriptor High DW (vme_MMUADD[1:0] = “11” - “10”)			
Bit[]	**Field**	**Description**	**Comments**
[63:32]	int_ADD[63:32]	Internal High Address remapping over 64-bit field. 64-bit addressing is mandatory for PCI Express addressing	PCI_Express recommendation to be able to support 64-bit addressing
Page Descriptor Low DW (vme_MMUADD[1:0] = “01” - “00”)			
Bit[]	**Field**	**Description**	**Comments**

<table>
<tr><th colspan="4">Page Descriptor High DW (vme_MMUADD[1:0] = “11” - “10”)</th></tr>
<tr><td>[0]</td><td>Page_Enable</td><td>Enable the Page entry. If not set, the incoming cycle is rejected, even if the VME address window is decoded.</td><td>This facility allows to built selective addressing</td></tr>
<tr><td>[1]</td><td>Write_Enable</td><td>Enable Write access across this page entry. If not set, the WRITE are rejected. (BERR# generation)
<b>Note</b> <i>This control allows to built WRITE protected area in the PEV_1100 resources. (Local SMEM or PCI Express remote resources)</i></td><td>Allows to support selected write-protect area</td></tr>
<tr><td>[4:2]</td><td>Traffic_Class[2:0]</td><td>PCI Express TC field. 2.2.6.6
<b>Note</b> <i>Only meaningful with remote PCI Express access.</i></td><td></td></tr>
<tr><td>[5]</td><td>Supervisory</td><td>While set only transactions with AM code Supervisory are decoded.</td><td></td></tr>
<tr><td>[7:6]</td><td>Byte Swapping</td><td>BYTE Swapping Policy. This 2-bit field specifies the Byte Swapping policy.(Little – Big Endian conversion)
<table>
<tr><td>00</td><td>No Swapping</td></tr>
<tr><td>01</td><td>LITTLE to BIG Endian Automatic Swapping
BYTE   : No Swapping
WORD : D16 word swapping
DW   : BYTE & WORD Swapping</td></tr>
<tr><td>10</td><td>DW BYTE & WORD Swapping</td></tr>
<tr><td>11</td><td>Reserved</td></tr>
</table></td><td>The Byte swapper is located in the VME Interface. Following control applies to the VME Slave data path</td></tr>
<tr><td>[11:8]</td><td>Space_Typ[3:0]</td><td>This 4-bit field encodes the Address Space targeted in destination. Refer to table “Address Type Encoding”</td><td></td></tr>
<tr><td>[15:12]</td><td>SW_Destination</td><td>Destination Address Space Type. 4-bit field
<table>
<tr><td>0000</td><td>PCI Express</td></tr>
<tr><td>0001</td><td>Reserved, Not allowed)</td></tr>
<tr><td>0010</td><td>Shared Memory</td></tr>
<tr><td>0011</td><td>User</td></tr>
<tr><td>Others</td><td>Reserved, Invalid value on actual PEV_1100</td></tr>
</table></td><td>Only two LSB bits used in the PEV_1100 implementation.</td></tr>
<tr><td>[17:16]</td><td>READ_Prefetch</td><td>Determine Destination prefetch size in case of VME64X BLT and MBLT. No used for 2eVME and 2eSST.
<table>
<tr><td></td><td>VME BLT</td><td>VME MBLT</td></tr>
<tr><td>00</td><td>32 Bytes</td><td>64 Bytes</td></tr>
<tr><td>01</td><td>64 Bytes</td><td>128 Bytes</td></tr>
<tr><td>10</td><td>128 Bytes</td><td>256 Bytes</td></tr>
<tr><td>11</td><td>256 Bytes</td><td>512 Bytes</td></tr>
</table></td><td>Only for VME64X addressing without information on data read size expected.</td></tr>
<tr><td>[18]</td><td>No_Snoop</td><td>Refer to PCI Express specification.</td><td>Only while targeting PCIe EP</td></tr>
<tr><td>[19]</td><td>Relaxed Ordering</td><td>Refer to PCI Express specification.2.2.6.4</td><td>Only while targeting PCIe EP</td></tr>
</table>

Page Descriptor High DW (vme_MMUADD[1:0] = “11” - “10”)			
[31:20]	int_ADD[31:20]	Internal Low Address remapping. 2.2.6.5	

Table 3.5. : Private VME MMU Page Descriptor

1780 The following table sums-up the destination Address Space Type . (PCI Express, USER or Shared Memory)

Space_Type	*PCI Express*	*Shared Memory / USER*
0x0-0x2	Reserved	Reserved
0x3-0x7	Reserved	Reserved
0x8	MEM A64	OK
0x9-0xF	Reserved	Reserved

Table 3.6 : Address Type Encoding

3.3.7 Private VME_Address Error Register

This register provides the catches VME Address stored while the VME Master interface received a 1785 BERR#.

PVME_ADDERR IO_Bus : 0x418-0x41B CR/CSR : 0x7F018-0x7F01B		**VME Address Error Register**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[1:0]	SWAP	R	0	Swapping Control bit.	
[31:2]	vme_ADD[31:2]	R	0	Current VME Address used while the VME BERR# acknowledge was received.	

3.3.8 Private VME_Status Error Register

This register provides the catched STATUS informations stored while the VME Master interface received a BERR#.

Reading the PVME_STAERR Register automatically rearms the VME Error catching logic.

PVME_STAERR IO_Bus : 0x41C-0x41F CR/CSR : 0x7F01C-0x7F01F		**VME Status Error Register**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[3:0]	TYPE	R	0x0	4-bit field encoding the VME AM. 0000 = Configuration	

PVME_STAERR IO_Bus : 0x41C-0x41F CR/CSR : 0x7F01C-0x7F01F		VME Status Error Register			
				0001 = A16 0010 = A24 0011 = A32 : SLT 0100 = A32 : BLT 0101 = A32 : MBLT 0110 = A32 : 2eVME 0111 = A32 : 2eSST 1111 = IACK Others = Reserved	
[8:4]	Length_DW	R	0x0	Block length in DW. Initial value issued by the PCI Express TLP	
[25:9]	Requester_ID	R	0x0	PCI Express Requester Identifier. Only Top section catched.	
[27:26]	Source_ID	R	0x0	Central Switch Port transaction source 00 = PCI Express EP Agent. 10 = IDMAC Agent. 11= USER Agent.	
[28]	BTO	R	0	VME Bus Time OUT Error. (Only while on-board BTO logic is active ... Slot_1 enabled)	
[29]	Write	R	0	VME transaction direction	
[30]	VME_Error_Over	R	0	Set while a 2nd error was detected while VME_Error_FL was already set.	
[31]	VME_Error_Flag	R	0	Error detected. While set, a VME BERR (Read or Write) was detected. Catcher parameter are related to the 1st error occurrence. This status flag is also connected to the VME ITC, allowing to issued an Interrupt.	

3.3.9 Private VME_LOCMON Register

The PEV_1100 implements 4 Location Monitors. A base register PVME_LOCMON holds a 32-bit address field A[31:5] and two control bit for the mode of operation. When a VME cycle match the specified Address (A[31:5] for A32, A[23:5] for A24 or A[15:5] for A16) a local interrupt is issued , selected by Address[4:3] field.

- PVME_LOCMON Register
  - [31:5] = Address
  - [4:3] = Location Monitor selection (1 to 4)
  - [2] = Reserved
  - [1:0] = VME Address Space as : Disable, A16, A24, A32

The four Location_Monitor interrupts are wired to the local ITC16 VME Interrupt controller. The PEV_1100 can issue VME cycles for location monitors with ADO or self addressed regular VME SLT cycles.

<table>
<tr><td colspan="2"><b>PVME_LOCMON</b>
IO_Bus : 0x420-0x423
CR/CSR : 0x7F020-0x7F023</td><td colspan="4"><b>VME Location Monitor Register</b></td></tr>
<tr><th>Bit[]</th><th>Function</th><th>R/W</th><th>Reset</th><th>Description</th><th>Comments</th></tr>
<tr><td>[1:0]</td><td>LocMON_ENA</td><td>RW</td><td>00</td><td>Location Monitor Enable. This 2-bit field enable the location Monitor in selected Address space.
= 00 : Disable
= 01 : VME A16
= 10 : VME A24
= 11 : VME A32</td><td></td></tr>
<tr><td>[4:2]</td><td>Reserved</td><td>R</td><td>000</td><td>Read only field.
<b><i>Note</i></b> <i>The 4 Locations Monitors are mapped with VME Address [4:3]</i></td><td></td></tr>
<tr><td>[31:5]</td><td>vme_ADD[31:5]</td><td>RW</td><td>0</td><td>VME 32-bit Location monitor Address
LocMON_ENA = 01 : vme_ADD[15:5]
10 : vme_ADD[23:5]
11 : vme_ADD[31:5]</td><td>VME Address mapping of the 4 Location Monitor</td></tr>
</table>

3.3.10 Private VME_LOCK Register

The PEV_1100 implements a dedicated register to support the VME AHOH + LOCK function.

1805

<table>
<tr><td colspan="2"><b>PVME_LOCK</b>
IO_Bus : 0x424-0x427
CR/CSR : 0x7F040-0x7F027</td><td colspan="4"><b>VME ADOH & LOCK Register</b></td></tr>
<tr><th>Bit[]</th><th>Function</th><th>R/W</th><th>Reset</th><th>Description</th><th>Comments</th></tr>
<tr><td>[0]</td><td>ADO_Mode</td><td>WR</td><td>0</td><td>When set, the scatter-gather VME Address Type 0xD and 0xE generate VME ADOH (in place of ADO)
<table>
<tr><th>ADO_Mode</th><th>= '0'</th><th>= '1'</th></tr>
<tr><td>Space_Type = 0xD</td><td>ADO:A24 (AM =0x35)</td><td>ADOH:A24 (AM =0x35)</td></tr>
<tr><td>Space_Type = 0xE</td><td>ADO:A32 (AM =0x05)</td><td>ADOH:A32 (AM =0x05)</td></tr>
</table></td><td>New capability to generate ADOH VME AM code, required to control LOCK process.</td></tr>
<tr><td>[1]</td><td>LOCK_Enable</td><td>WR</td><td>0</td><td>When set, VME Master LOCK process is enabled</td><td>Enable the FSM LOCK support.</td></tr>
<tr><td>[2]</td><td>LOCK_CLEAR_Mode</td><td>WR</td><td>0</td><td>This control bit allows to select the LOCK Clear operation mode :
= 0 : LOCK Cleared with LOCK_CLEAR_Cmd = '1'. (IO_Space path)
= 1 : LOCK Cleared with ADOH/ADO Write or with LOCK_CLEAR_Cmd = '1'. (IO_Space path)</td><td>Support for LOCK clear command.</td></tr>
</table>

<table>
<tr><th colspan="2">PVME_LOCK
IO_Bus : 0x424-0x427
CR/CSR : 0x7F040-0x7F027</th><th colspan="4">VME ADOH & LOCK Register</th></tr>
<tr><td>[3]</td><td>LOCK_COMP_Mode</td><td>WR</td><td>0</td><td>This control bit allows to select the VME WRITE# signal polarity driving
= 0 : VME WRITE# is de-asserted while VME LOCK ADOH transaction is process.
= 1 : VME WRITE# is asserted while VME LOCK ADOH transaction is process</td><td></td></tr>
<tr><td>[4]</td><td>LOCK BTO_Ena</td><td>WR</td><td>0</td><td>LOCK Time out Enable.
= 1 LOCK remains active for maximum.. 1[ms]
= 0 LOCK remains active until specific LOCK_CLEAR activated</td><td></td></tr>
<tr><td>[6:5]</td><td>Reserved</td><td>R</td><td>0</td><td>Not used</td><td></td></tr>
<tr><td>[7]</td><td>LOCK_CLEAR_Cmd</td><td>W</td><td>0</td><td>LOCK flag Clear Command.</td><td></td></tr>
<tr><td>[10:8]</td><td>LOCK_Status[2:0]</td><td>WR</td><td>000</td><td>LOCK Status
<table>
<tr><th>LOCK Status</th><th></th></tr>
<tr><td>“000”</td><td>LOCK Cleared</td></tr>
<tr><td>“100”</td><td>LOCK WON (ADOH with DTACK#)</td></tr>
<tr><td>“101”</td><td>LOCK In progress (ADOH with RETRY#)</td></tr>
<tr><td>“001”</td><td>LOCK ERROR (ADOH with BERR#)</td></tr>
<tr><td>“010”</td><td>LOCK ERROR (Error detected during DATA phase)</td></tr>
<tr><td>“011”</td><td>LOCK ERROR (Time out 1 [ms]) only when “LOCK BTO_Ena” = '1'</td></tr>
</table></td><td>Directly attached to the LOCK FSM</td></tr>
<tr><td>[11]</td><td>LOCK_DATA_Error</td><td>WC</td><td>0</td><td>VME LOCK Error detected</td><td></td></tr>
</table>

3.3.11 Private VME_RMW Registers

The PEV_1100 implements four dedicated registers to support the VME RMW atomic transactions. Refer to chapter 2.11.11 for RMW support.

PVME_RMW_MODE IO_Bus : 0x428-0x42B CR/CSR : 0x7F028-0x7F02B		VME RMW Mode + Cycle Management			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[1:0]	RMW_Size_Mode[1:0]	RW	00	VME RMW Cycle OP Size	

PVME_RMW_MODE IO_Bus : 0x428-0x42B CR/CSR : 0x7F028-0x7F02B	**VME RMW Mode + Cycle Management**				
				00 Reserved (Not suported) 01 VME RMW D08 10 VME RMW D16 11 VME RMW D32	
[4:0]	RMW_ADD_Mode[2:0]	RW	000	VME RMW Cycle Address Space 000 VME A16 DATA USER 001 VME A16 DATA SUP 010 VME A24 DATA USER 011 VME A24 DATA SUP 100 VME A32 DATA USER 101 VME A32 DATA SUP others Reserved	
[27:6]	Reserved	R	0	Not used	
[30:28]	RMW_Status	R	000	This status field provide real-time execution status of current/previous RMW cycle. 000 VME cycle is ended 100 RMW Cycle executed with BERR 010 RMW Cycle executed with Compare Not_OK 001 RMW Cycle executed with Compare OK others Reserved	
[31]	RMW_Trigger	W	0	This control bit allows to start the VME RMW transaction	
	RMW_Cycle_RUN	R	0	This status bit reflects the VME RMW execution 0 VME cycle is ended 1 VME RMW cycle in execution	

PVME_RMW_ADD IO_Bus : 0x42C-0x42F CR/CSR : 0x7F02C-0x7F02F	**VME RMW Address**				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**

PVME_RMW_ADD IO_Bus : 0x42C-0x42F CR/CSR : 0x7F02C-0x7F02F		**VME RMW Address**			
[31:0]	RMW_Address31:0]	RW	00	VME RMW Address	

PVME_RMW_CMP IO_Bus : 0x430-0x433 CR/CSR : 0x7F030-0x7F033		**VME RMW Data Compare**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	RMW_CMPDAT[31:0]	RW	00	VME RMW Data Compare . This data field is used by the Read cycle of the RMW.	

PVME_RMW_UPT IO_Bus : 0x434-0x437 CR/CSR : 0x7F034-0x7F037		**VME RMW Data Update**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	RMW_UPTDATC[31:0]	RW	00	VME RMW Data Update. This data field is used by the Write cycle of the RMW.	

3.3.12 Private VME_ELBDIR Register

Reserved register for compatibility with IPV_1102. It is dedicated to the P2020 ELB → VME Master low latency bridge

<table>
<tr><th colspan="2"><b>PVME_ELBDIR</b>
IO_Bus : 0x43C-0x43F
CR/CSR : 0x7F03C-0x7F03F</th><th colspan="4"><b>VME P2020 ELB - VME Master Direct</b></th></tr>
<tr><th>Bit[]</th><th>Function</th><th>R/W</th><th>Reset</th><th>Description</th><th>Comments</th></tr>
<tr><td>[2:0]</td><td>VDIR_AMMODE[2:0]</td><td>RW</td><td>0</td><td>This data field determine the VME AM code used during the VME Master Direct access from the P2020 ELB.
<table>
<tr><th>AMMODE</th><th>AM</th><th>Description</th></tr>
<tr><td>000</td><td>0x29</td><td>A16 Data User</td></tr>
<tr><td>001</td><td>0x2D</td><td>A16 Data Supervisor</td></tr>
</table></td><td></td></tr>
</table>

<table>
<tr><td colspan="2"><b>PVME_ELBDIR</b>
IO_Bus : 0x43C-0x43F
CR/CSR : 0x7F03C-0x7F03F</td><td colspan="4"><b>VME P2020 ELB - VME Master Direct</b></td></tr>
<tr><td></td><td></td><td></td><td></td><td><table><tr><td>010</td><td>0x39</td><td>A24 Data User</td></tr><tr><td>011</td><td>0x3D</td><td>A24 Data Supervisor</td></tr><tr><td>100</td><td>0x09</td><td>A32 Data User</td></tr><tr><td>101</td><td>0x0D</td><td>A32 Data Supervisor</td></tr><tr><td>110</td><td>0x3F</td><td>IACK Cycle</td></tr><tr><td>111</td><td>0x3F</td><td>Reserved</td></tr></table></td><td></td></tr>
<tr><td>[3]</td><td>VDIR_D32MODE</td><td>RW</td><td>0</td><td>This control bit enable the VME64x D32 access. While set, the two ELB transactions forming the D32 operand access are concatenated to issue a single VME D32 access.
= '0' : ELB transactions are handled individually
= '1' : ELB transactions are handled in 32-bit Operand</td><td></td></tr>
<tr><td>[5:4]</td><td>VDIR_ELBSIZ[1:0]</td><td>RW</td><td>0</td><td>This field define the ELB GPCM window size allocated for the VME Master Direct Access.<table><tr><th>ELBSIZ</th><th>Description</th></tr><tr><td>00</td><td>16 MBytes</td></tr><tr><td>01</td><td>32 MBytes</td></tr><tr><td>10</td><td>64 MBytes</td></tr><tr><td>11</td><td>256 MBytes</td></tr></table></td><td></td></tr>
<tr><td>[23:6]</td><td>Reserved</td><td>R</td><td>0</td><td>Not used</td><td></td></tr>
<tr><td>[31:24]</td><td>VDIR_ADDH[31:24]</td><td>RW</td><td>0x00</td><td>VME High Address. The field provides VME MSB Address required while VME A32 is used.
<b><i>Note</i></b> This field can be reduced in case of the ELB window is extended over 16 MBytes.</td><td></td></tr>
</table>

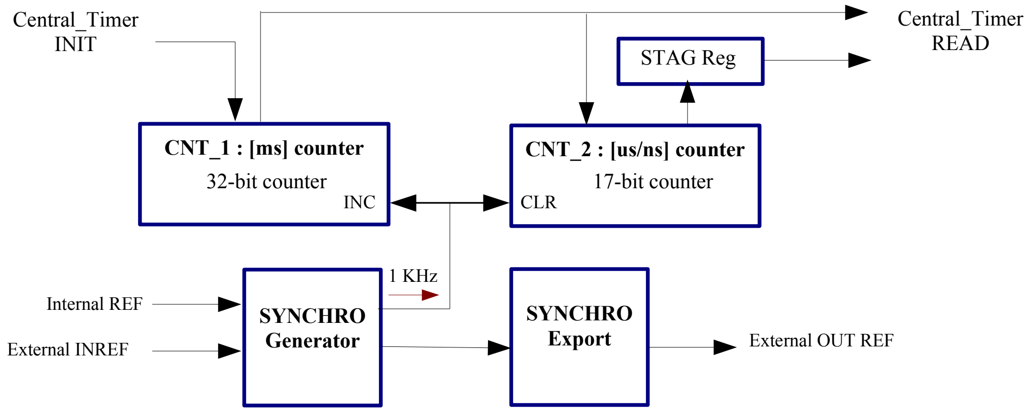
3.3.13 Private VME_GLOBTIM Register

Many simulation require a centralized time base, used for data time-stamping. The PEV_1100 Central_Timer is implemented around two counter-timers and a Master/Slave synchronization mechanisms.

- CNT_1 as a 32-bit 1.000[ms] counter (or optionally 1.024 [ms])
- CNT_2 as a 17-bit pre-counter selectable 1, 5, 25 or 125 MHz, providing precise time inside the [ms] quantum of CNT_1.

The PEV_1100 Global Timer can issues local interrupts derived from the the 1.000[ms] synchronization

Four registers mapped on the IO_Bus are made available to support the Global Timer service. These registers can be access from the PCI Express IO Space or from the VME CRCSR/A24 decoded area.



The external synchronization on the VME Bus is supported on SYSFAIL# or with IRQ#1 / IRQ#2. The PEV_1100 can be programmed as Synchronization Master or as Synchronization Slave. Provision is made to support external synchronization (IRIG-B and/or GPS) . FPGA user's area can be assigned to implement such mechanism.

The PVME_GLTIM_CSR holds main control and status information.

PVME_GLTIM_CSR IO_Bus : 0x440-0x443 CR/CSR : 0x7F040-0x7F043		**VME Global Timer CSR**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[1:0]	gtim_PRE	RW	00	Global Timer CNT_2 counter precision. This 2-bit filed select statically the pre-scaling to support : 00 = 1 MHz (1 [us]) 01 = 5 MHz (200 [ns]) 10 = 25 MHz (40 [ns]) 11 = 100 MHz (10 [ns])	CNT_2 is 17-bit. 1025*125 = 128'000 = 0x1F400 Max CNT_2 count = 0x1FFFF = 1310 [us]
[2]	Reserved	R	0	Not used	
[3]	gtim_1024	RW	0	Global Timer CNT_1 time base mode. 0 = 1000 [us]	The 1024 [us] mode is provided for legacy

PVME_GLTIM_CSR IO_Bus : 0x440-0x443 CR/CSR : 0x7F040-0x7F043	**VME Global Timer CSR**				
				1 = 1024 [us]	compatibility with TSI 148
[6:4]	gtim_SYNC_SRC	RW	000	Global Timer Synchronisation Source. This 3-bit field select the source of the local synchronization. 000 = Local time reference. Based on local 100[ppm] 125 MHz oscillator. 001 = User Source #1 010 = User Source #2 100 = VME SYSFAIL# high to low front edge. 101 = VME IRQ#1 high to low front edge. 110 = VME IRQ#2 high to low front edge. Others = Reserved. **Note** *User Source are reserved for dedicated logic based on external synchronisation as GPS or IRIG-B.* **Note** *Synchronization based on VME IRQ#1 or IRQ#2 introduce an additional 250[ns] jitter.*	
[7]	gtim_LOC_ENA	RW	0	Local Time base reference Enable. When set, the on-chip local timer 1000/1024 [us] is activated. While enabled it can provide the local Top Synchronisation reference.	
[9:8]	gtim_OUT_SEL	RW	000	Global Timer Master Synchronization Selection. This 2-bit field allows to select VME signals used as Master Synchronization. While activated, a normalized 8 [us] pulse is issued every 1000/1024 [us] on selected VME line. 00 = No VME Master Synchronisation 01 = VME SYSFAIL# 10 = VME IRQ1# 11 = VME IRQ2#	Due to VME TTL Open collector signalling, the synchronization is related to the front edge HIGH to LOW transition.
[15:10]	Reserved	R	000	Not implemented	
[16]	gtim_SYNC_ERR_FL	WC	0	Global Timer Synchronization Error. Flag. While set an error was detected. Refer to CNT_2 register. Error Flag Clear Command. Writing '1' clear the error flags located in the CNT_2	
[19:17]	gtim_FSM	R	000	Reserved for IOxOS Verification	
[30:20]	Reserved	R	000	Not implemented	
[31]	gtim_ENABLE	RW	0	Global Timer Enable. When '0', the complete Global Timer function is deactivated.	

The PVME_GLTIM_CNT2 holds the low part of the global timer with associated synchronization monitoring.

PVME_GLTIM_CNT2 IO_Bus : 0x448-0x44B CR/CSR : 0x7F048-0x7F04B	VME Global Timer LOW Counter				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[16:0]	gtim_CNT_2	RW	00	Global Timer CNT_2. Max 17-bit with counting rate determined by gtim_PRE[1:0]. (1, 5, 25 or 100 MHz)	Whatever frequency selected, the intrinsic precision remains unchanged (50 [ppm])
[19:17]	Reserved	R	0	Not Implemented	
[27:20]	Synchro_DELTA	R	0	This 8-bit field provides the synchronization derived time, at latest TOP synchronization (in 8[ns]quantum increment) Value provided is from -124 to +125 : 0b1000'0111 = - 928[ns] error (early) 0b1111'1111 = - 8[ns] error (early) 0b0000'0000 = perfect synchronization 0b0000'0001 = + 8[ns] error (late) 0b0111'1101 = + 928[ns] error (late)	This field allows to qualify the external synchronization related to the internal clock.
[28]	Synchro_PRE_ERR	R	0	Error status flag. The TOP synchronisation has arrived before the qualified window (T0 - 928[ns]).	The qualified window is fixed as +/- 926[ns], corresponding of ~ 1000[ppm] precision.
[29]	Synchro_POST_ERR	R	0	Error status flag. The TOP synchronisation has arrived after the qualified window. (T0 + 928[ns]).	
[30]	Synchro_ROLLOVER	R	0	Error status flag. No TOP synchronisation arrived before the 17-bit counter roll-over. When it is occurring the Global Timer function is disabled.	
[31]	Synchro_ERROR	R	0	OR of the three previous Error status flags.	

The PVME_GLTIM_CNT1 holds the 32-bit main counter. This support ~ 1200 hours without roll over.

PVME_GLTIM_CNT1 IO_Bus : 0x44C-0x44F CR/CSR : 0x7F04C-0x7F04F	VME Global Timer Main Counter				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	gtim_CNT_1	RW	00	Global Timer CNT_1. 32-bit timer-counter attached to the TOP Synchronization. (1[ms])	gtim_CNT_1 can be initialized at specific value.

To support coherent CNT_2 + CNT_1 read-out, CNT-1 shall be read first. A dedicated logic copy in a stag register the CNT_2 value during the CNT-1 read-out.

3.3.14 Private VME_ITC Controller

A dedicated Interrupt controller ITC is embedded in the VME64x section. 16 Interrupt Source are connected to the ITC16.

The following table sums up the VME64x Interrupt sources allocation.

ITC INT#	Mode	Assignation	Comments
#0	LEVEL	VME SYSRESET# assertion	Cassidian specific
#1	LEVEL	VME IRQ_1	Legacy VME Interrupt. Auto vectorization enabled
#2	LEVEL	VME IRQ_2	
#3	LEVEL	VME IRQ_3	
#4	LEVEL	VME IRQ_4	
#5	LEVEL	VME IRQ_5	
#6	LEVEL	VME IRQ_6	
#7	LEVEL	VME IRQ_7	
#8	LEVEL	VME ACFAIL#	
#9	LEVEL	VME_Error (Master)	VME64x Cycle Error
#10	LEVEL	VME Central Time #1	Global Time 1000/1024[ms] tic
#11	LEVEL	VME Central Time #2	Global Time Error flag
#12	LEVEL	Location Monitor #0	
#13	LEVEL	Location Monitor #1	
#14	EDGE	Location Monitor #2	
#15	EDGE	Location Monitor #3	

Table 3.7. : Private VME ITC Interrupt Sources

3.3.15 VME64x CR/CSR ADER_0 Register

This register, formed by four consecutive BYTE Registers, defines the VME A32 Base Address for subsequent PEV_1100 Slave mapping. The window size is defined with **vslv_WSIZ[2:0]**, located in **PVME_SLVCSR** Register.

The PEV_1100 can only be mapped in the A32 Address Space. VME A64 addressing is not supported.

VMECSR_ADER_0_3 IO_Bus : 0x560 CR/CSR : 0xF'FF63		**CR/CSR ADER_0 MSB Address Decoder Compare**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[7:0]	A32_MAP[31:24]	R	0	Address offset, mapping the VME Slave. The window is programmable from 1 to 2048 MBytes. Only valid bit are significant.	

VMECSR_ADER_0_2 IO_Bus : 0x564 CR/CSR : 0xF'FF67	CR/CSR ADER_2 Address Decoder Compare				
Bit[]	Function	R/W	Reset	Description	Comments
[3:0]	Reserved	R	0	Not used..	Refer to ANSI/VITA 1.1 VME64 Extensions specification
[7:4]	A32_MAP[23:20]	R	0	Address offset, mapping the VME Slave. The window is programmable from 1 to 2048 MBytes. Only valid bit are significant.	

VMECSR_ADER_0_1 IO_Bus : 0x568 CR/CSR : 0xF'FF6B	CR/CSR ADER_1 Address Decoder Compare				
Bit[]	Function	R/W	Reset	Description	Comments
[7:0]	Reserved	R	0	Not used..	Refer to ANSI/VITA 1.1 VME64 Extensions specification

VMECSR_ADER_0_0 IO_Bus : 0x56C CR/CSR : 0xF'FF6F	CR/CSR ADER_1 LSB Address Decoder Compare				
Bit[]	Function	R/W	Reset	Description	Comments
[5:0]	AM[5:0]	R	0	Not implemented..	
[6]	DFSR	R	0	Dynamic Function Size Read. While set, the window size defined by vslv_WSIZ[2:0] is provided in the "A32_MAP[31:20]" field.	This function allows to determine the VME A32 Slave window size required by the PEV-1100
[7]	XAM Mode	R	0	Not implemented..	

1865 ***Note*** *The complete ADER register is built with four Byte section allocate in consecutive DWORD Register.*

Note *The VME Slave interface shall also be enabled with specific control bit located in the* ***PVME_SLVCSR*** *Register.*

3.3.16 VME64x CR/CSR BSR & BCR Registers

1870 These two Registers, allows to SET or CLEAR control bits individually. A '1' written to the selected Control bit will set the function in the BSR Register and Clear it the BCR Register.

- BSR = Bit Set Register
- BCR = Bit Clear Register

VMECSR_BCR/BSR IO_Bus : 0x5F4 / 0x5F8 CR/CSR : 0xF'FFF7 / ... B	**CR/CSR BSR BIT Set Register** **CR/CSR BSR BIT Clear Register**				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[7]	PV_RESET	RW	0b0	PEV_1100 RESET Mode. While SET, the PEV_1100 is forced in RESET Mode	
[6]	PV_SYSFAIL_Driver	RW	0b0	PEV_1100 VME SYSFAIL Driver Enable	
[5]	PV_SYSFAIL	RW	0b0	PEV_1100 VME SYSFAIL Direct Control	
[4]	PV_ENABLE	RW	0b0	PEV_1100 VME Slave Enable.	
[3]	PV_BERR_Flag	RW	0b0	PEV_1100 VME BERR condition generated.	Flag information related for exception handling.
[2]	CRAM_OWNER	R	0	Not implemented.	
[1:0]	Reserved	R	0	Not used.	

1875

3.3.17 VME64x CR/CSR BAR Register

This register provides status information of PEV_1100 CR/CSR Base Address .

VMECSR_BAR IO_Bus : 0x5FC CR/CSR : 0xF'FFFF	**CR/CSR Base Address Register**				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[7:3]	CR/CSR_Base[23:19]	R	0	VME CR/CSR Base Address. Dependant on the VME 64x Mode Switch. -ON- = GA Geographic Address -OFF- = Switch Group_A[5:1]	Refer to VME64 Specification
[2:0]	Reserved	R	0	Not used..	Refer to VME64 Specification

3.3.18 VME64x CR Configuration ROM

1880 The PEV_1100 implements the Configuration ROM. A 2 KBytes ROM stores the VME64X parameters and is made available through the CR space.

3.4 Shared_Memory & IDMAC CSR Resources

1885 The SMEM-IDMA CSR Resources mapping is provided in following table.

Offset IBUS	Size	Register_Name	Description	Comments
0x800 - 0x803	0x4	**SMEM_DDR2_CSR**	Four port DDR2 Controller Control & Status	
0x804 - 0x807	0x4	**SMEM_DDR2_ERR**	DDR2 Addressing Error status	
0x840 - 0x843	0x4	**IDMA_GCSR**	IDMA Global CSR	
0x850 - 0x843	0X4	**IDMA_RD_0_PCSR**	Pipeline Register	
0x880 - 0x88F	0x10	**IDMA_ITC**	Intelligent DMA & Shared Memory Interrupt Controller	
0x900 - 0x903	0x4	**IDMA_RD_0_CSR**	IDMA Read Engine #0 Control&Status	
0x904 - 0x907	0x4	**IDMA_RD_0_NDES**	IDMA Read Engine #0 Next DMA_Descriptor Pointer	
0x908 - 0x90b	0x4	**IDMA_RD_0_CDES**	IDMA Read Engine #0 Current DMA_Descriptor Pointer	
0x90c - 0x90F	0x4	**IDMA_RD_0_WCNT**	IDMA Read Engine #0 WCNT and DMA Operation Status	
0x940 - 0x94F	0x10	**IDMA_RD_1_XXX**	IDMA Read Engine #1 Registers	
0xA00 - 0xA0F	0x10	**IDMA_WR_0_XXX**	IDMA Write Engine #0Registers	
0xA40 - 0xA4F	0x10	**IDMA_WR_1_XXX**	IDMA Write Engine #1 Registers	

Table 3.8. : SMEM – IDMA IO_Bus resource

3.4.1 SMEM_DDR2 Register

This register provides control and status informations related to the local Shared Memory DDR2 Controller.

SMEM_DDR2_CSR IO_Bus : 0x800 - 0x803		**DDR2 SMEM Control & Status**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[0]	ddr2_REFRATE	RW	0	DDR2 Refresh Rate control. = 0 : 7.8[us]. For commercial temperature up to 70 °C = 1 : 3.9[us]. Required in extended temperature range up to 85 °C	
[1]	ddr2_REFMODE	RW	0	DDR2 Refresh Mode control. = 0 : DDR2 Refresh executed one at the time = 1 : DDR2 Refresh executed in burst of eight consecutive.	
[3:2]	ddr2_SIZE[1:0]	R		PEV implemented DDR2 Size. = 01 : 256 MBytes = 10 : 512 MBytes = 11 : 1 GBytes.	

SMEM_DDR2_CSR IO_Bus : 0x800 - 0x803		**DDR2 SMEM Control & Status**			
[7:4]	ddr2_ARBMODE[1:0]	RW	00	DDR2 Multi port access policy. The arbitration policy sharing the DDR2 memory array is determined by : = 0000 : Round Robin with READ priority over WRITE. = 0001 : Round Robin with READ and WRITE. as same priority level. = 0010 : High priority on Port_A (Shared Memory mapped on the Central Switch). Others : Reserved for application specific RT policy.	
[11:8]	ddr2_MASK[3:0]	RW	0x0	Individual Port masking. Allows to dynamically Enable/Disable the READ and WRITE pending request to the DDR2 Memory Controller.	
[12]	RD_Cache_ENA	RW	0	READ Ahead cache Global Enable. While set, the 4 KBytes READ ahead logic is activated.	
[13]	DDR2_Ctr_ENA	RW	1	DDR2 Controller Global Enable. While set, the DDR2 controller is activated. This control bit can be RESET to force the DDR2 inactive.	Reserved for DDR2 debugging.
[15:14]	Reserved				
[16]	MREADY	R	0	DDR2 Controller Ready. The DDR2 initialization process is ended and is ready to accept READ or WRITE transaction.	
[17]	DLL_Locked	R	0	DDR2 associated DCM (DLL) is locked. OK	Status information on FPGA macro function
[18]	DLY_Ready	R	0	DDR2 IO Delay controller is ready OK	
[23:19]	Reserved	RW	0	Not implemented	
[31:24]	Reserved	RW			

3.4.2 SMEM_DDR2_ERR Register

1890 This register provides flag status informations related to reported Error in Shared Memory DDR2 Controller.

SMEM_DDR2_ERR IO_Bus : 0x804 - 0x807		**DDR2 SMEM Error**			
Bit[]	**Function**	**R/W**	**Rese t**	**Description**	**Comments**
[0]	WR_Address_Error	RWc	'0'	Addressing Error on Write access	
[1]	RD_Address_Error	RWc	'0'	Addressing Error on Read access	
[31:02]	Reserved	R	'0'	Not used	

3.5 PEV Intelligent DMA Controller

1895 A single global IDMA register provides general control and status. Each IDMA Engine owns four control & status registers, mapped on the IO_Bus infrastructure. following table sums up the current implementation.

Register	Register description	Read_0	Read_1	Write_0	Write_1
GCSR	Global IDMA CSR	0x840			
PCSR	Trigger Pipeline	0x850	0x854	0x858	0x85C
CSR	General Control & Status Information	0x900	0x940	0xA00	0xA40
NDES	NEXT DMA_Descriptor pointer. 32-bit Address to local DDR2 SMEM	0x904	0x944	0xA04	0xA44
CDES	CURRENT DMA_Descriptor pointer.	0x908	0x948	0xA08	0xA48
DCNT	DMA DW counter and runtime status	0x90c	0x94c	0xA0c	0xA4c

Table 3.9. : IDMA Registers mapping

Provision is made to extend the DMAC subsystem to eight engines. Future implementation could need additional DMA channels.

3.5.1 IDMA_GCSR Register

1900 This register provides general control and status informations, related to the complete DMA function. (READ_Engines and WRITE_Engines)

IDMA_GCSR IO_Bus : 0x840-0x843	**IDMA Global GCSR**				
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[0]	Read_Response_ERRFF	RWc	0	Read_Response error detected. This flag remains set until clear with write '1' over it. • Corrupted Completion TLB (Read_response • Bad Completion TLB sequence • Unexpected Completion TLB	
[1]	Read_Timeout_ERRFF	RWc	0	Read_Response time-out detected. This flag remains set until clear with write '1' over it. • No Read_Response after 500[us]. Current TLP is aborted.	
[2]	Reserved	R	0	Not used	Reserved
[3]	Read_Response_STA[3]	RWc	0	TLP Header Invalid.	
[4]	Read_Response_STA[4]	RWc	0	TLP Tag Signature Invalid.	
[5]	Read_Response_STA[5]	RWc	0	TLP Acknowledge Type unsupported.	
[6]	Read_Response_STA[6]	RWc	0	DMA Channel not active.	
[7]	Read_Response_STA[7]	RWc	0	DMA Channel No Read Request pending.	
[29:8]	Reserved	R	0	Not used	
[30]	Read_Timeout_ENA	RW	1	Read-Request – Read-Response 5000[us] time-	

IDMA_GCSR IO_Bus : 0x840-0x843		**IDMA Global GCSR**			
				out enable. The time-out logic can be disabled	
[31]	Global_IDMA	RW	1	When '1', enable the four IDMA Engines	

3.5.2 IDMA_XX_N_PCSR Register

These registers provide general control and status informations related to the DMA Pipeline operation. Each RD_Engine or WR_Engine own a dedicated PCSR Register.

IDMA_XX_N_PCSR IO_Bus : 0x850 RD_0 IO_Bus : 0x854 RD_1 IO_Bus : 0x858 WR_0 IO_Bus : 0x85C WR_1		**IDMA Engine Pipeline Trigger TCSR**			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[5:0]	Trigger_PIPE_CNT[5:0]	R	0x00	6-bit Trigger IN Pipeline counter. Each Engine owns this event counter, but are managed differently for RD_Engine and WR_Engine. For RD_Engine T rigger_PIPE_CNT • INC on Trigger_IN condition • DEC on DMA_Descriptor execution For WR_Engine T rigger_PIPE_CNT • INC on DMA_Descriptor execution • DEC on Trigger_IN condition	New IDMA mode of operation allowing to pipeline WR_Engine → RD_Engine with embedded ring-buffer management.
[7:6]	Trigger_PIPE_MAX[1:0]	RW	0b00	Define WR_Engine maximal ring-buffer entries. Condition the DMA_Descriptor start execution. 00 4 entries 01 8 entries 10 16 entries 11 32 entries	Not used for RD_Engine
[9:8]	Trigger_INSEL[1:0]	RW	00	Trigger_IN Counter increment selection : 00 RD_Engine #0 Trigger_OUT 01 RD_Engine #1 Trigger_OUT 10 WR_Engine #0 Trigger_OUT 11 WR_Engine #1 Trigger_OUT	
[15]	Trig_PIPE_Enable	RW	0	Trigger_PIPE Enable = 0 : Disable IDMA Trigger_IN Pipeline (CNT) = 1 : Enable IDMA Trigger_IN Pipeline (CNT) While enabled, the IDMA Start_Mode is conditioned by the value of the	

IDMA_XX_N_PCSR — **IDMA Engine Pipeline Trigger TCSR**
IO_Bus : 0x850 RD_0
IO_Bus : 0x854 RD_1
IO_Bus : 0x858 WR_0
IO_Bus : 0x85C WR_1

				Trigger_PIPE_CNT	
[31:16]	Reserved	R	0x0000	Not used	

3.5.3 IDMA_XX_N_CSR Register

This register provides general control and status informations related to the DMA Execution

IDMA_RD_0_CSR — **IDMA RD_Engine #N Control & Status**
IO_Bus : 0x900 - 0x903

Bit[]	Function	R/W	Reset	Description	Comments
[0]	des_ext_PORT	R	0	DMA Descriptor DW0 status information	
[2:1]	ext_ADD_Mode[1:0]	R	00	DMA Descriptor DW0 status information	
[3]	ext_RDERR	R	0	DMA Descriptor DW0 status information	
[4]	ext_SNOOP	R	0	DMA Descriptor DW0 status information	
[5]	ext_RELAX	R	0	DMA Descriptor DW0 status information	
[7:6]	ext_AT	R	00	DMA Descriptor DW0 status information	
[9:8]	smem_ADD_Mode[1:0]	R	00	DMA Descriptor DW0 status information	
[11:10]	trig_OUTGEN	R	00	DMA Descriptor DW0 status information	
[13:12]	Update_MOD[1:0]	R	00	DMA Descriptor DW0 status information	
[15:14]	Reserved	R	00	Not used	
[17:16]	ext_ADDBND	R	00	DMA Descriptor DW0 status information (Address boundary)	
[19:18]	ext_PSIZ	R	00	DMA Descriptor DW0 status information (Packet size)	
[21.20]	idma_CH_NB	R	TBC	DMA Channel Number (0 to 3)	IDMA Channel Signature information
[22]	idma_RDENG	R	TBC	DMA READ Engine	
[23]	idma_WRENG	R	TBC	DMA WRITE Engine	
[26:24]	idma_FSM_DESMGT	R	000	This 3-bit field provides current state information of the IDMA “EXEC” State Machine. 000 Idle 001 DMA Descriptor fetch Request 010 Update DMA Descriptor fetched 011 Wait for trigger to start the DMA Descriptor execution. 100 DMA Descriptor execution in progress. 101 Next DMA Descriptor evaluation (Continue, Stop or Abort) 110 Current DMA Descriptor Update in	

IDMA_RD_0_CSR IO_Bus : 0x900 - 0x903		IDMA RD_Engine #N Control & Status			
				progress. (Write back GTIOM and Status) 111 Reserved	
[27]	idma_CACHE	RW	0	DMA Cache Enable. When set, the local DPRAM cache DMA_Descriptor is enabled	
[28]	idma_SUSPEND	RW	0	DMA Channel SUSPEND, while set the DMA activity is stopped and resumed while the control bit is reset..	
[29]	idma_ABORT	RW	0	The current DMA Channel activity is aborted. The Read process is immediately stopped but data already in the DPRAM buffer are written to the destination. ***WARNING*** *This control shall be SET-RESET under software control.*	
[30]	idma_KILL	RW	0	The current DMA Channel activity is immediately stopped and data in the local DPRAM buffer are flushed. ***WARNING*** *This control shall be SET-RESET under software control.*	
[31	idma_ENABLE	RW	0	The DMA channel is enabled and ready to receive a 1st load in the IDMA_XX_N_NDES Register (Start of chain). This global control also enable the chaining of following DMA_descriptor	

3.5.4 IDMA_XX_N_NDES Register

This register provides the address pointer of the next DMA_Descriptor. At the end of execution of the current IDMA process, this pointer will be used to fetch next DMA_Descriptor.

IDMA_RD_0_NDES IO_Bus : 0x904 - 0x907		IDMA RD_Engine #n Next DMA_Descriptor Pointer			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[2:0]	N_START	RW		Next DMA Descriptor START Option. Determine how the DMA_Descriptor will begin its execution.	
[3]	Reserved	RW		Not used	
[4]	C_LAST	RW		Current DMA Descriptor LAST Indicator	
[31:5]	N_DMA_Desc[31:5]	RW		Next DMA Descriptor SMEM address pointer	

3.5.5 IDMA_XX_N_CDES Register

This register provides the address pointer of the current (in progress) DMA_Descriptor. This register is a copy of previous one “IDMA_XX_N_NDES” while a new DMA_Descriptor is loaded.

IDMA_RD_0_CDES IO_Bus : 0x908 - 0x90B		IDMA RD_Engine #n Current DMA_Descriptor Pointer			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[2:0]	C_START	R		Current DMA Descriptor START Option. Determine how the DMA_Descriptor has begun its execution.	
[3]	Reserved	R		Not used	
[4]	C_LAST	R		Useless	
[31:5]	C_DMA_Desc[31:5]	R		Current DMA Descriptor SMEM address pointer	

3.5.6 IDMA_XX_N_DCNT Register

This register provides information on current DMA process. This includes the current DW word counter and complementary real-time status.

IDMA_RD_0_DCNT IO_Bus : 0x90C - 0x90F		IDMA RD_Engine #n DMA DW Counter			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[0]	idma_Error_WRITE	R	0	Status flag information	
[1]	idma_Error_READ	R	0	Status flag information	
[23:2]	dma_WCNT[23:2]	R		Current DMA DW word counter.	
[25:24]	idma_FSM_RDCTL	R	00	This 2-bit field provides current state information of the IDMA READ controller State Machine. 00 Idle 01 READ Evaluate best OP 10 READ transaction running 11 READ process is terminated, Wait for end of WRITE process. Need to empty FIFO buffer.	
[27:26]	idma_FSM_WRCTL	R	00	This 2-bit field provides current state information of the IDMA READ controller State Machine. 00 Idle 01 WRITE Evaluate best OP 10 WRITE transaction running 11 WRITE process is terminated.	
[28]	Reserved	R		Not used	
[31:29]	idma_STATUS[2:0]	R	000	Real-time DMA operation status 000 Idle 001 1st DMA_Descriptor on wait ... 010 DMA In Progress (Running even	

IDMA_RD_0_DCNT IO_Bus : 0x90C - 0x90F	IDMA RD_Engine #n DMA DW Counter

	Descriptor)
011	DMA In Progress (Running odd Descriptor)
100	DMA Ended OK.
101	DMA Ended KILL/ABORT.
110	DMA Ended ERROR_Read/Write.
111	DMA Ended ERROR with out of range SMEM addressing

3.5.7 SMEMIDMA Block Interrupt Allocation

This SMEM IDMA Block ITC16 controller manage following interrupt sources

- IDMA Write Engine Normal END / ERROR
- IDMA Read Engine Normal END / ERROR

Following table sums up the USER Block Interrupt source allocation

ITC INT#	Mode	Assignation	Comments
#0	LEVEL	IDMA Read Engine #0 Normal END	
#1	LEVEL	IDMA Read Engine #0 ERROR	
#2	LEVEL	IDMA Read Engine #1 Normal END	
#3	LEVEL	IDMA Read Engine #1 ERROR	
#4	LEVEL	IDMA Write Engine #0 Normal END	
#5	LEVEL	IDMA Write Engine #0 ERROR	
#6	LEVEL	IDMA Write Engine #1 Normal END	
#7	LEVEL	IDMA Write Engine #1 ERROR	
#8	LEVEL	Not used	
#9	LEVEL	Not used	
#10	LEVEL	Not used	
#11	LEVEL	Not used	
#12	LEVEL	Not used	
#13	LEVEL	Not used	
#14	EDGE	Not used	
#15	EDGE	Not used	

Table 3.10. : IDMA ITC Interrupt Sources

3.5.8 DMA_Descriptor Element

The DMA Engines (READ and WRITE) executes chained list of DMA_Descriptor. Each DMA_Descriptor owns a complete sub DMA operation description. Same chained list of DMA_Descriptor can be executed by the Read and Write Engine. The DMA Engines fetch and executes only DMA_Descriptor located in the Shared Memory (256 to 1024 MBytes)

Each DMA_Descriptor is composed of eight consecutive 32-bit words (32 Bytes) aligned on 32 Bytes boundary. The first six DW are only read by the DMA engines and the 7th and 8th DW can also be write back with update to the SMEM Memory.

DW_0 : Main Control

DW_1 : Word Counter and Address Management

DW_2 : Shared Memory Address pointer.

DW_3 : Next Descriptor Address pointer.

DW_4 : External Address Low pointer.

DW_5 : External Address High pointer.

DW_6 : Signature and Time Stamping UPDATE

DW_7 : 32-bit Time stamping from Global Timer UPDATE (c.f VME64X Interface)

Following table provides complete DMA_Descriptor description.

Ass.	Name	Description	Comments	Note
DW_0 Main Engine Control				
0 : <0>	**ext_Port**	Select the DMA DIRECT Port 0 Normal, Central SWITCH port 1 DIRECT Port	Only available for IDMA Channel #0	
0 : <2:1>	**ext_ADD_Mode[1:0]**	External Address (64-bit) Address Control. This 2-bit field support : 00 Normal Update. 01 No Update. Previous ext_ADD[63:2] is used as the new address pointer 10 No_Increment. (with normal Update) 1x Reserved	The No_Increment Mode supports FIFO based interface read-out.	
0 : <3>	**ext_DMAERR**	External DMA Error Bypass 0 Abort current Operation, Finish current write data and stop the DMA chaining 1 + Continue with next DMA_Descriptor		
0 : <4>	**ext_SNOOP**	PCI Express Target Memory Snooping Enable / Disable. Defined by the PCI Express	Used to control the Cache coherency policy	

<table>
<tr><th>Ass.</th><th>Name</th><th>Description</th><th>Comments</th><th>Note</th></tr>
<tr><td></td><td></td><td>specification.</td><td></td><td></td></tr>
<tr><td>0 : &lt;5&gt;</td><td><b>ext_RELAX</b></td><td>PCI Express Relax ordering. READ transaction tagged with ext_RELAX are authorized to pass pending WRITE. Defined by the PCI Express specification.</td><td>Refer to PCI Express 2.0 Specification.</td><td></td></tr>
<tr><td>0 : &lt;7:6&gt;</td><td><b>ext_AT[1:0]</b></td><td>PCI Express Address Translation Mode. Defined by the PCI Express specification.</td><td>Refer to PCI Express 2.0 Specification.</td><td></td></tr>
<tr><td>0 : &lt;9:8&gt;</td><td><b>smem_ADD_Mod[1:0]</b></td><td>Shared Memory Address Mode. This 2-bit field select the Shared Memory Address Pointer update operation.
<table>
<tr><td>00</td><td>Normal Update. smem_ADD[31:2] is used as the new pointer</td></tr>
<tr><td>01</td><td>No_Update. The smem_ADD[31:2] is not used. Current value, usually defined by previous DMA_Descriptor</td></tr>
<tr><td>10</td><td>Add_Update. The smem_ADD[31:2] is added to the current value.</td></tr>
<tr><td>1x</td><td>Reserved</td></tr>
</table></td><td>The No_Update mode allows to built/consume continuous data structure in the Shared Memory, with external non-continuous data.
The Add_Update mode allows to implement generic DMA_Descriptor for data collection. 1st element can define the start of data structure and subsequent with relative offset.</td><td></td></tr>
<tr><td>0 : &lt;11:10&gt;</td><td><b>trig_OUTGEN</b></td><td>This 2-bit field selects the idma_TRIG_OUT[n] generation.
<table>
<tr><td>00</td><td>No generation</td></tr>
<tr><td>01</td><td>Reserved</td></tr>
<tr><td>10</td><td>TRIG_OUT generated at Start of DMA_Descriptor execution .</td></tr>
<tr><td>11</td><td>TRIG_OUT generated at End of DMA_Descriptor execution .</td></tr>
</table></td><td>Each Engine own its idma_TRIG_OUT[n] field encoding :</td><td></td></tr>
<tr><td>0 : &lt;13:12&gt;</td><td><b>Update_Mode[1:0]</b></td><td>DMA_Descriptor Update Policy. This 2-bit field determines the related policy.
<table>
<tr><td>00</td><td>No Update DMA Descriptor DW6&amp;7</td></tr>
<tr><td>10</td><td>Update with :
<b>DW6</b> = Signature_ID[7:0] + Spare[7:0] + GTIM Low
<b>DW7</b> = GTIM High (1KHz)</td></tr>
<tr><td>11</td><td>Update with :
<b>DW6</b> = DMA_Error[3:0] + Signature_ID[7:0] + WCNT[23:2] + DMA_Error[1:0]
<b>DW7</b> = GTIM High (1KHz)</td></tr>
<tr><td>01</td><td>Reserved</td></tr>
</table></td><td>At the end of a DMA Descriptor execution, the two last DW of currently executed DMA Descriptor can be updated (write-back) with specific informations as :
• Global Time
• Current WNCT value
• DMA Error Status flag</td><td></td></tr>
<tr><td>0 : &lt;15:14&gt;</td><td><b>Interrupt_mode[1:0]</b></td><td>Interruption generation control. This 2-bit field determines how/when related interrupt is issued.</td><td>Interrupt can be programmed for each DMA_Descriptor.</td><td></td></tr>
</table>

Ass.	Name	Description	Comments	Note
		00 Interrupt disabled 01 Interrupt End generated at start of execution of current DMA_Descriptor 10 Interrupt End generated at end of execution of current DMA_Descriptor. 11 Interrupt Error + End generated at end of execution of current DMA_Descriptor.		
0 : <17:16>	**ext_ADDBND[1:0]**	External SWITCH Boundary size. This 2-bit field shall specify address boundary crossing rule. 00 4 KBytes (PCI Express rule) 01 2 KBytes VME64x 2eVME+2eSST 10 1 KBytes VME MBLT/BLT64 11 256 Bytes VME BLT/BLT32	By default, PCI Express rule is selected. In case of limited field determined by the VME64X ext_Type[3:0] parameter, the ext_ADDBND[1:0] field shall be set according.	
0 : <19:18>	**ext_PSIZ[1:0]**	External SWITCH maximum Packet size. 00 128 Bytes 01 256 Bytes 10 512 Bytes 11 1 KBytes	On actual INTEL/AMD chip-set implementation, the PCI Express, MAXPAYLOADSIZE is limited to 128 or 256 Bytes. This can have a bad impact on the PCI Express bandwidth performance.	
0 : <22:20>	**ext_TC[2:0]**	External PCI Express Traffic Class. Defined by the PCI Express specification.	PCI Express QoS. Limited implementation. Need multi-VC ...	
0:<23>	**Reserved**	Not used		
0 : <25:24>	**max_OUT_RR**	Maximal Outstanding Read_Request (Only applicable for WR_Engine) 00 1* Read_Request 01 2* Read_Request 10 3* Read_Request 11 4* Read_Request (To be verified)	Allow to control, per DMA Descriptor, the Read_Request pipeline depth. Remote Host System memory read will require pipeline on Read_Request (multiple issued) to optimize the PCI Express performance	
0 : <27:26>	**dst_SWAP**	Destination Swapping policy. This 2-bit field implements the 2-bit swapping control used by the Destination Agent_SW	Used only for VME64x (big endian) interface.	

Ass.	Name	Description	Comments	Note
		00 No Swapping 01 LITTLE to BIG Endian Automatic Swapping BYTE : No Swapping WORD : D16 word swapping DW : BYTE & WORD Swapping 10 DW BYTE & WORD Swapping 11 DW swapping in QW access		
0 : <31:28>	Reserved	Not used		
DW_1 Word Counter				
1 : <1:0>	Reserved	Not used, fixed to b"00"	Only DW DMA count	
1 : <23:2>	**dma_WCNT**	DMA Size. 22-bit DW counter allowing to built DMA operation over 16 MBytes – 4.		
1 : <26:24>	**SW_Agent**	Agent_SW Destination/Source Address 000 PCI Express EP 001 VM64X 010 DDR2 Shared Memory 011 USER 1xx Reserved	For READ_Engine, defines the Agent_SW Destination. For WRITE_Engine, defines the Agent_SW Source.	
1 : <27>	Reserved	Reserved for larger Central_SW implementation	PEV_1100 implement a 4-ports Central_SW.	
1 : <31:28>	**Ext_Type[3:0]**	Agent_SW Destination Address TYPE, used to encode specific addressing mode	i.e VME AM	
DW_2 Shared Memory Address Pointer				
2 : <1:0>	**smem_RQoS[1:0]**	Local SMEM Read maximum burst size. 00 2 KBytes 01 1 KBytes 10 512 Bytes 11 256 Bytes	Capability to limit the READ bursting size.	
2 : <31:2>	**smem_ADD[31:2]**	Shared Memory Address Pointer	This field can also be used	
DW_3 Next DMA_Descriptor Pointer				
3: <2:0>	**Start_Mode[2:0]**	Current DMA Descriptor Execution Start. This 3-bit field defined the execution beginning of current DMA Descriptor. 000 Immediate Execution. 001 Wait on next Global Time synchronization 010 Wait on GPIO Event #1 (To be defined) 011 Wait on GPIO Event #2 (To be defined)	While the current DMA_Descriptor is fetched, its execution start is conditioned by following field. The Start Trigger on Global Timer TIC allows to define DMA Engine Descriptor chain attached to this particular event.	

Ass.	Name	Description	Comments	Note
		000 – Immediate Execution. 100 – Wait on local DMA Trigger_IN #1 event or if Trigger_PIPE_Enable= '1', wait on • Trigger_PIPE_CNT /= "000000" for RD_Engine • Trigger_PIPE_CNT <= "100000" for WR_Engine 101 – Wait on local DMA Trigger_IN #2 event 110 – Wait on local DMA Trigger_IN #3 event 111 – Wait on local DMA Trigger_IN #4 event	GPIO Events, #1, #2 are reserved for external trigger condition defined in the USER area. The local DMA Trigger #1,2,3,4 are controlled directly by the DMA Engine.	
3: <3>	Reserved			
3: <4>	**Last_DESC**	Last Descriptor Execution. While set, the current DMA_Descriptor chain is stopped. 0 – Continue DMA Descriptor execution. 1 – Last DMA Descriptor, STOP after current execution		
3: <31:5>	**Next_ADD**	Shared Memory Next DMA_Descriptor pointer		
DW_4 External Low Address (Address pointer to VME64x, PCI Express and USER Central_SW Agent)				
4: <1:0>	Reserved			
4: <31:2>	**Ext_ADD[31:2]**	LSB External Address Pointer		
DW_5 External High Address (Address pointer to VME64x, PCI Express and USER Central_SW Agent)				
5 : <31:0>	**Ext_ADD[63:32]**	MSB External Address pointer.	The high address section is not incremented on Low Address roll-over. Software shall handle the 4 GBytes roll-over	
DW_6 Global Time				
6a: <16:0>	**G_Time_CNT2[16:0]**	High speed Global timer section, providing intra millisecond time information. Frequency/precision can be selected from 1, 5, 25 or 125 MHz.	The precision selection is determined with control bits located in the VME64x section.	(1)
6a: <23:17>	Reserved	Not used		
6a : <31:24>	**Sequence_ID[7:0]**	This 8-bit field is read with the DMA_Descriptor fetch and write back, incremented by 1, at the end of the current DMA_Descriptor execution.	Debugging facilities allowing to track the DMA_Descriptor execution.	
6b: <23:0>	**Current_WCNT[23:0]**	Current word counter WCNT		
6b : <31:24>	**Sequence_ID[7:0]**	Idem as 6a		
DW_7 Global Time 32-bit 1 [ms] synchronized				
7 : <31:0>	**G_Time_CNT1[31:0]**	32-bit Global Timer millisecond counter timer. At the end of the current DMA_Descriptor execution, a copy of the Global Timer is write back to the Shared Memory, at location of the original	The write-back capability to SMEM can be enabled/disabled with control field in 1st DW	(1)

Table 3.11.: IDMA Descriptor Format

1940

3.6 USER CSR Resources

The USER CSR Resources are mapped on the IO_Bus interface as :

- IO_Bus 1 KBytes Device#4 (0xC00-0xFFF)

1945 The USER CSR Registers & Resources implements control and status registers for following functions :

- GPIO programmable control
- Message Passing FIFO

1950 The following tables sums up the USER Private Register Mapping.

Offset IO_BUS	Size	Register_Name	Description	Comments
0xC00 - 0xC03	0x4	**GPIO_A_OUT**	GPIO Group_A OUTPUT Register	
0xC04 - 0xC07	0x4	**GPIO_A_ENA**	GPIO Group_A TRI-STATE Register	
0xC08 - 0xC0B	0x4	**GPIO_A_IN**	GPIO Group_A INPUT Register	
0xC0C - 0xC0F	0x4	**GPIO_A_POL**	GPIO Group_A POLARITY Interrupt Register	
0xC10 - 0xC13	0x4	**GPIO_B_OUT**	GPIO Group_B OUTPUT Register	
0xC14 - 0xC17	0x4	**GPIO_B_ENA**	GPIO Group_B TRI-STATE Register	
0xC18 - 0xC1B	0x4	**GPIO_B_IN**	GPIO Group_B INPUT Register	
0xC1C - 0xC1F	0x4	**GPIO_B_POL**	GPIO Group_A POLARITY Interrupt Register	
0xC20 - 0xC23	0x4	**Msg_FIFO_CTL_0**	Message FIFO#0 Control & Status	Cassidian specific
0xC24 - 0xC27	0x4	**Msg_FIFO_CTL_1**	Message FIFO#1 Control & Status	Cassidian specific
0xC28 - 0xC2B	0x4	**Msg_FIFO_CTL_2**	Message FIFO#2 Control & Status	Cassidian specific
0xC2C - 0xC2F	0x4	**Msg_FIFO_CTL_3**	Message FIFO#3 Control & Status	Cassidian specific
0xC30 - 0xC33	0x4	**Msg_FIFO_Port_0**	Message FIFO#0 Control & Status IO Port	Cassidian specific
0xC34 - 0xC37	0x4	**Msg_FIFO_Port_1**	Message FIFO#1 Control & Status IO Port	Cassidian specific
0xC38 - 0xC3B	0x4	**Msg_FIFO_Port_2**	Message FIFO#2 Control & Status IO Port	Cassidian specific
0xC3C - 0xC3F	0x4	**Msg_FIFO_Port_3**	Message FIFO#3 Control & Status IO Port	Cassidian specific
0xC40 - 0xC7F	0x30	**Reserved**	Not implemented	
0xC80-0xC8F	0x10	**USER_ITC**	USER ITC Interrupt Pending	Refer to ITC chapter
0xC90 - 0xF7F	0x30	**Reserved**	Not implemented	
0xF80 - 0xFFF	0x80	**Signature_ROM**	Defined ROM area for software signature	

Table 3.12. : USER Register Description (Cassidian MAS specific)

Note *This default mapping could be replaced any time by custom specific implementation.*

3.6.1 User GPIO_A Output Register

This register allows to define the Output level on GPIO[32:1] (Port_A)

GPIO_A_OUT IO_Bus : 0xC00 - 0xC03		GPIO Group_A OUTPUT Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	GPIO_A_OUT[31:0]	RW	0	Define the OUTPUT level for each GPIO Group_A Control Bit <0> : GPIO[1] ... <31> : GPIO[32]	

3.6.2 User GPIO_A Enable Register

This register allows to define the direction (Input or Output) on GPIO[32:1] (Port_A)

GPIO_A_ENA IO_Bus : 0xC04 - 0xC07		GPIO Group_A ENABLE Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	GPIO_A_ENA[31:0]	RW	0	Define the DIRECTION for each GPIO Group_A. While '1' defined as Output, '0' as Input. <0> : GPIO[1] ... <31> : GPIO[32]	

3.6.3 User GPIO_A Input Register

This register allows to read the level on GPIO[32:1] (Port_A)

GPIO_A_IN IO_Bus : 0xC08 - 0xC0B		GPIO Group_A INPUT Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	GPIO_A_IN[31:0]	R	0	Direct status information <0> : GPIO[1] ... <31> : GPIO[32]	

3.6.4 User GPIO_A INT POLARITY Register

This register allows to control the polarity of GPIO_A_IN[3:0] send to the ITC16 Interrupt controller.

GPIO_A_POL IO_Bus : 0xC08 - 0xC0B		GPIO Group_A INT POLARITY Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**

GPIO_A_POL IO_Bus : 0xC08 - 0xC0B		GPIO Group_A INT POLARITY Register			
[3:0]	GPIO_A_POL[3:0]	RW	0	Direct status information <0> : GPIO_POL[1] ... <3> : GPIO_POL[3]	Cassidian specific
[31:4]	Reserved	R	0	Not used	

3.6.5 User GPIO_B Output Register

This register allows to define the Output level on GPIO[64:33] (Port_B)

GPIO_B_OUT IO_Bus : 0xC0C - 0xC0F		GPIO Group_B OUTPUT Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	GPIO_A_OUT[31:0]	RW	0	Define the OUTPUT level for each GPIO Group_B Control Bit <0> : GPIO[33] ... <31> : GPIO[64]	

3.6.6 User GPIO_B Enable Register

This register allows to define the direction (Input or Output) on GPIO[64:33] (Port_B)

GPIO_B_ENA IO_Bus : 0xC14 - 0xC17		GPIO Group_B ENABLE Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	GPIO_B_ENA[31:0]	RW	0	Define the DIRECTION for each GPIO Group_B. While '1' defined as Output, '0' as Input. <0> : GPIO[33] ... <31> : GPIO[64]	

3.6.7 User GPIO_B Input Register

This register allows to read the level on GPIO[64:33] (Port_B)

GPIO_B_IN IO_Bus : 0xC18 - 0xC1B		GPIO Group_B ENABLE Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	GPIO_A_IN[31:0]	R	0	Direct status information <0> : GPIO[33] ...	

GPIO_B_IN IO_Bus : 0xC18 - 0xC1B		GPIO Group_B ENABLE Register			
				<31> : GPIO[64]	

3.6.8 User GPIO_B INT POLARITY Register

This register allows to control the polarity of GPIO_A_IN[3:0] send to the ITC16 Interrupt controller.

GPIO_B_POL IO_Bus : 0xC1C - 0xC1F		GPIO Group_B INT POLARITY Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[3:0]	GPIO_A_POL[3:0]	RW	0	Direct status information <0> : GPIO_POL[1] ... <3> : GPIO_POL[3]	Cassidian specific
[31:4]	Reserved	R	0	Not used	

1970 3.6.9 Message FIFO#N Control Registers

These four register provides control and status for the four Message Passing FIFO resources.

MSG_FIFO_CTL#N IO_Bus : 0xC30-0xC33 IO_Bus : 0xC34-0xC37 IO_Bus : 0xC38-0xC3B IO_Bus : 0xC3C-0xC3F		Message Passing FIFO#N Control Register Offset +0x0 → FIFO#0 Offset +0x4 → FIFO#1 Offset +0x8 → FIFO#2 Offset +0xC → FIFO#3			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[7:0]	MSG_FIFO_WCNT[7:0]	R	0x00	Current Word counter. Provides the number of DW stored in the FIFO	Cassidian specific
[15:8]	MSG_FIFO_WRPT[7:0]	R	0x00	Current SRAM Write Pointer	Cassidian specific
[23:16]	MSG_FIFO_RDPT[7:0]	R	0x00	Current SRAM Read Pointer	Cassidian specific
[24]	MSG_FIFO_NotEMPTY	R	0b0	FIFO Not_EMPTY flag. Set while WNCT /= 0x00	Cassidian specific
[25]	MSG_FIFO_FULL	R	0b0	FIFO FULL flag. Set while WNCT = 0xFF	Cassidian specific
[26]	Reserved	R	0b0	Not used	
[27]	MSG_FIFO_ERRF	RW	0b0	This flag status is positioned to '1' while a FIFO operation error is detected • WRITE to FIFO FULL. • Read from FIFO EMPTY This flag can be clear by writing '1' over it.	Cassidian specific
[28]	MSG_FIFO_RESET	RW	0b0	Message Passing FIFO RESET. While written to '1' the FOFO control is reset. WCNT, WRPT and RDPT forced to 0x00.	Cassidian specific
[29]	MSG_FIFO_TMEM_REA	RW	0b0	TOSCA MEMORY Access Message Passing FIFO read enabled.	Cassidian specific

MSG_FIFO_CTL#N IO_Bus : 0xC30-0xC33 IO_Bus : 0xC34-0xC37 IO_Bus : 0xC38-0xC3B IO_Bus : 0xC3C-0xC3F				**Message Passing FIFO#N Control Register** Offset +0x0 → FIFO#0 Offset +0x4 → FIFO#1 Offset +0x8 → FIFO#2 Offset +0xC → FIFO#3	
[30]	MSG_FIFO_TMEM_WEA	RW	0b0	TOSCA MEMORY Access Message Passing FIFO write enabled.	Cassidian specific
[31]	MSG_FIFO_ENA	RW	0b0	Global Message Passing FIFO Enable. Applicable in both accessing pathway (IO_Bus and TMEM)	Cassidian specific

3.6.10 Message FIFO IO Port

Each Message Passing FIFO own a single access port supporting Read and Write access. Erroneous access as Read FIFO Empty or Write FIFO Full

MSG_FIFO_PORT#N IO_Bus : 0xC20-0xC23 IO_Bus : 0xC24-0xC27 IO_Bus : 0xC28-0xC2B IO_Bus : 0xC2C-0xC2F				**Message Passing FIFO#N IO Port** Offset +0x0 → FIFO#0 Offset +0x4 → FIFO#1 Offset +0x8 → FIFO#2 Offset +0xC → FIFO#3	
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[31:0]	Message[31:0]	RW	--	Message port 32-bit WRITE → FIFO Load READ → FIFO Retrieve	Cassidian specific

3.6.11 USER Block Interrupt Allocation

This USER Block ITC16 controller manage following interrupt sources

- Message passing FIFO flags FULL and NotEMPTY
- GPIO inputs
- Write Posting Error

Following table sums up the USER Block Interrupt source allocation

ITC INT#	Mode	Assignation	Comments
#0	LEVEL	MSG_FIFO#0_NotEMPTY	
#1	LEVEL	MSG_FIFO#1_NotEMPTY	
#2	LEVEL	MSG_FIFO#2_NotEMPTY	
#3	LEVEL	MSG_FIFO#3_NotEMPTY	
#4	LEVEL	MSG_FIFO#0_FULL	
#5	LEVEL	MSG_FIFO#1_FULL	
#6	LEVEL	MSG_FIFO#2_FULL	
#7	LEVEL	MSG_FIFO#3_FULL	

#8	LEVEL	GPIO_A_0 (PEV_1100 → Jn14_01)	
#9	LEVEL	GPIO_A_1 (PEV_1100 → Jn14_02)	
#10	LEVEL	GPIO_B_0 (PEV_1100 → Jn14_32)	
#11	LEVEL	GPIO_B_1 (PEV_1100 → Jn14_33)	
#12	LEVEL	GPIO_B_2 (PEV_1100 → Jn14_34)	
#13	LEVEL	GPIO_B_3 (PEV_1100 → Jn14_35)	
#14	EDGE	Message Passing FIFO Write Posted Error	
#15	EDGE	IO_Bus Bridge Write Posted Error	

Table 3.13. : USER Block ITC Interrupt Sources

3.6.12 User Signature_ROM

This area provides a non-volatile information related to the USER Block.

- Device_ID and Revision_ID.
- Any valuable software related information.

Signature_ROM IO_Bus : 0xF80 - 0xFFF		**Standard ROM area**			
Location	**Function**	**R/W**	**Reset**	**Description**	**Comments**
0x0:[31:0]		R	--	16-bit Function_ID 16-bit Revision_ID	
0x1:[31:0]				To Be defined	
0x1F:[31:0]		R	--	Checksum	

3.7 ITC Interrupt Controller

The ITC16 Interrupt controller owns four control registers. The PEV_1100 instantiates four ITC16 (one in each Agent_SW)

Following table sums up the ITC16 resource mapping.

ITC Register	Local/PCIe	VME64x	SMEM/IDMAC	USER
IACK	0x080	0x480 (Reserved)	0x880 (Reserved)	0xc80 (Reserved)
CSR	0x084	0x484	0x884	0xC84
IMC	0x088	0x488	0x888	0xC88
IMS	0x08c	0x48c	0x88c	0xC8c

Table 3.14.: ITC Register IO_Bus mapping

In normal operation, only IACK#0 register is used. The “PCI Express MSI Generator” manages under hardware control the selection of the appropriate ITC IACK Register.

3.7.1 ITC IACK Register

This register provides Interrupt vectorization on Read and selectable IP source clear on Write. Writing to the ITC_IACK register also restart the IP scanning, and therefore re-enable the Interrupt Controller.

ITC_IACK offset + 0x0		Interrupt Acknowledge Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[7:0]	Status_ID[7:0]	R	0x00	With AUTO_VEC = '0' : 0x00 With AUTO_VEC = '1' : VME Status_ID	AUTO_VEC only available on ITC VME.
[11:8]	IP_Src[3:0]	R	0x0	Encoded Interrupt Source. 0X0 is INT#0	
[13:12]	ITC_Src[1:0]	R	0b00	ITC Controller Source 00 : Local / PCI Express 01 : VME64X 10 : SMEM / IDMA 11 : USER	
[14]	Reserved	R	0b0	Not implemented	
[15]	vec_VALID	R	--	IACK Vector Valid	
[31:16]	IP_CLR[15:0]	W	--	Selective IP Clear	
[31:16]	IP[15:0]	R	--	Provide Interrupt Pending status information	

3.7.2 ITC CSR Register

This register provides general control & status informations.

ITC_CSR offset + 0x4		Interrupt Control & Status Register			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[0]	GLENA	R/W	0b0	Global Enable 0 = ITC is disable, no MSI Request are issued 1 = ITC is enable. Unmasked IP issues	
[1]	GLCLR_IP	W	0b0	Command to clear all Interrupt pending IP	
[2]	AUTO_VEC	R/W	0b0	VME Automatic IACK. When set, on VME INT detection a VME IACK is automatically issued before. Only activated on VME ITC.	
[7:3]	Reserved	R		Not implemented	
[11:8]	Last_IPS[3:0]	R		Memorized last IP Served.	
[12]	Reserved	R		Not implemented	
[15:13]	IPC_State[2:0]	R		ITC FSM Status 000 Idle, 001 Scanning IP 010 MSI Request Pending, 011 MSI Request Acknowledge, 100 INT pending, wait for IACK 101 Wait on remote Clear IP 110 VME Auto-vector pending 111 VME Auto-vector executed	
[31:16]	IP_CLR[15:0]	W	--	Selective IP Clear	
[31:16]	IPM[15:0]	R	--	Provide Interrupt Pending Masked status information	IPM(i) = IP(i) and IM(i)

3.7.3 ITC IMC Register

This register provides direct Interrupt Mask CLEAR function.

ITC_IMC offset + 0x8		Interrupt Mask Clear			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[15:0]	IMC_[15:0]	R/WC	0xFFFF	Interrupt Mask Clear . Writing a bit '1', reset the selected Mask, Writing a bit '0' has no effect. Each bit is assigned to its relative location. Register READ provide Mask status = 0 : Interrupt Source is Enabled.(non masked) = 1 : Interrupt Source is Masked.	
[31:16]	IP[15:0]	R	--	Provide Interrupt Pending status information	

3.7.4 ITC IMS Register

This register provides direct Interrupt Mask SET function.

ITC_IMS offset + 0xC		Interrupt Mask Set			
Bit[]	**Function**	**R/W**	**Reset**	**Description**	**Comments**
[15:0]	IMS_[15:0]	R/WS	0xFFFF	Interrupt Mask Set . Writing a bit '1', set the selected Mask, Writing a bit '0' has no effect. Each bit is assigned to its relative location. Register READ provide Mask status = 0 : Interrupt Source is Enabled.(non masked) = 1 : Interrupt Source is Masked.	
[31:16]	IPM[15:0]	R	--	Provide Interrupt Pending Masked status information	IPM(i) = IP(i) and IM(i)

4 ANNEXES

4.1 Compressed 256 Bytes IO Space

To support PC BIOS with IO Space size limited to 256 bytes (as suggested by the PCI 3.0 Specification) a new PCI Express IO Space mapping has been defined.

This alternate IO Space mapping is enabled by positioning to '1' the mini-DIP switch SW502-2. (

Generic software can check the status of the switch SW502-2 in the “ILOC_STATIC” register to determine automatically the IO_Space mode activated.

The Compressed mode maps with following

- 1st region of thirty-two(32) key (most important) PEV_1100's registers
- 2nd region of Sixteen(16) registers supporting the four(4) ITC16 controllers
- 3rd egion of Sixteen(16) registers whose are r- allocable with a 2-bit selector field “Dyn_IOSEL[1:0]” located in the “ILOC_STATIC” register

Following tables provide the PCI Express IO_Space resource mapping while the “Compressed Mode” is enabled

a) Fixed section mapping 48 registers. (32 +16)

b) Dynamic section mapping four(4) times 16 registers

Offset IO_Space PCI	Size	Register_Name	Description	Comments
Key/Main Resources				
0x00 - 0x03	0x4	**ILOC_STATIC**	External Static Information (micro-switch & strapping)	Hold status information of static options
0x04 - 0x07	0x4	**ILOC_SIGN**	Signature FPGA date of Creation 32-bit field	
0x08 - 0x0B	0x4	**ILOC_GENCTL**	General Control Register	
0x0C - 0x0F	0x4	**PCIE_MMUADD**	PCI Express INgress MMU Address pointer.	
0x10 - 0x13	0x4	**PCIE_MMUDAT**	PCI Express INgress MMU Data Register	
0x14 - 0x17	0x4	**V5_SMON_ADDPT**	VIRTEX-5 System Monitor Address Pointer	
0x18 - 0x1B	0x4	**V5_SMON_DAT**	VIRTEX-5 System Monitor Data Register	
0x1C - 0x1F	0x4	**V5_SMON_STA**	VIRTEX-5 System Monitor Status	
0x20 - 0x23	0x4	**PVME_SLOT1**	VME64X Slot-1 related functions as Arbiter, BTO and Static option switches status.	
0x24 - 0x27	0x4	**PVME_MASCSR**	VME64X Master Port related control and status	
0x28 - 0x2B	0x4	**PVME_SLVCSR**	VME64X Slave Port related control and status	
0x2C - 0x2F	0x4	**PVME_INTG**	VME64X Interrupt Generator	
0x30 - 0x33	0x4	**PVME_MMUADD**	VME64x INgress MMU Address pointer.	
0x34 - 0x37	0x4	**PVME_MMUDAT**	VME64x INgress MMU Data Register	

Offset IO_Space PCI	Size	Register_Name	Description	Comments
0x38 - 0x3B	0x4	**PVME_LOCMON**	VME64x Location Monitor Address	
0x3C - 0x3F	0x4	**PVME_LOCK**	VME64x ADOH & LOCK support	VME64x ADOH & LOCK support
0x40 - 0x43	0x4	**SMEM_DDR2_CSR**	Four port DDR2 Controller Control & Status	
0x44 - 0x47	0x4	**IDMA_GCSR**	IDMA Global CSR	
0x48 - 0x4B	0x4	**IDMA_RD_0_PCSR**	IDMA Pipeline Control Register Read Engine #0	
0x4C - 0x4F	0x4	**IDMA_WR_0_PCSR**	IDMA Pipeline Control Register Write Engine #0	
0x50 - 0x53	0x4	**IDMA_RD_0_CSR**	IDMA Read Engine #0 Control&Status	
0x54 - 0x57	0x4	**IDMA_RD_0_NDES**	IDMA Read Engine #0 Next DMA_Descriptor Pointer	
0x58 - 0x5B	0x4	**IDMA_RD_0_CDES**	IDMA Read Engine #0 Current DMA_Descriptor Pointer	
0x5C - 0x5F	0x4	**IDMA_RD_0_WCNT**	IDMA Read Engine #0 WCNT and DMA Operation Status	
0x60 - 0x63	0x4	**IDMA_WR_0_CSR**	IDMA Write Engine #0 Control&Status	
0x64 - 0x67	0x4	**IDMA_WR_0_NDES**	IDMA Write Engine #0 Next DMA_Descriptor Pointer	
0x68 - 0x6B	0x4	**IDMA_WR_0_CDES**	IDMA Write Engine #0 Current DMA_Descriptor Pointer	
0x6C - 0x6F	0x4	**IDMA_WR_0_WCNT**	IDMA Write Engine #0 WCNT and DMA Operation Status	
0x70 - 0x73	0x4	**PVME_GLTIM_CSR**	Global Timer Control & Status Register	
0x74 - 0x77				
0x78 - 0x7B	0x4	**PVME_GLTIM_CNT2**	Global Timer Pre-scaler 17-bit	
0x7c - 0x7f	0x4	**PVME_GLTIM_CNT1**	Global Timer Main 32-bit counter (1 [ms]	
0x4C - 0x4F				
0x80 - 0x83	0x4	**PCIE_BE_ITC_IP**	PCIE_BE ITC Interrupt Pending	ITC16 Controller PCIE Agent_SW
0x84 - 0x87	0x4	**PCIE_BE_ITC_IACK**	PCIE_BE Local ITC Interrupt Acknowledge	
0x88 - 0x8B	0x4	**PCIE_BE_ITC_IMC**	PCIE_BE ITC Mask CLEAR Control Reg.	
0x8C - 0x8F	0x4	**PCIE_BE_ITC_IMS**	PCIE_BE ITC Mask SET Control Reg.	
0x90 - 0x9F	0x10	**PVME_ITC**	VME64x ITC Interrupt Controller	ITC16 Controller
0xA0 - 0xAF	0x10	**SMEMIDMA_ITC**	SMEM-IDMA ITC Interrupt Controller	ITC16 Controller
0xB0 - 0xBF	0x10	**USER_ITC**	USER ITC Interrupt Controller (optional)	ITC16 Controller
Multiplexed Resources				
0xC0 ...	0x4	**REFER TO FOLLOWING TABLES**		
...				
... 0xFF	0x4			

2030

SEL_Multiplexed = 00 → PCIE_EP Agent_SW

0xC0 - 0xC3	0x4	**ILOC_STATIC**	External Static Information (micro-switch & strapping)	Hold status information of SW SW502-2
0xC4 - 0xC7	0x4	**ILOC_CABLE_1**	External PCI Express Connector #1	
0xC8 - 0xCC	0x4	**ILOC_CABLE_2**	External PCI Express Connector #2	
0xCC - 0xCF	0x4	**ILOC_PONFSM**	Power_ON FSM Sequencer Status	
0xD0 - 0xD3	0x4	**ILOC_SPI**	SPI Flash EPROM Bit programming	
0xD4 - 0xD7	0x4	**ILOC_PCIE_SW**	PCI Express Switch PEX8624 Control & Status	
0xD8 - 0xDC	0x4	**PCIE_EPSTA**	PCI Express EP Status	
0xDC - 0xDF	0x4	**V5_PCIEP_ADDPT**	VIRTEX-5 PCI Express EP Address Pointer	
0xE0 - 0xE3	0x4	**V5_PCIEP_DAT**	VIRTEX-5 PCI Express EP Data Register	
0xE4 - 0xE7	0x4	**V5_PCIEP_SEL**	VIRTEX-5 PCI Express EP Status Selection	
0xE8 - 0xEC	0x4	**V5_PCIEP_RSLT**	VIRTEX-5 PCI Express EP Selection Result	
0xEC - 0xEF	0x4	**PMCXMC**	PMC & XMC Side-band Signalling	
0xF0 - 0xF3	0x4	**I2C_CTL**	I2C Controller Control Register	
0xF4 - 0xF7	0x4	**I2C_CMD**	I2C Controller Command Register	
0xF8 - 0xFC	0x4	**I2C_DATW**	I2C Controller Data Write Register	
0xFC - 0xFF	0x4	**I2C_DATR**	I2C Controller Data Read Register	

SEL_Multiplexed = 01→ VME64X Agent_SW

0xC0 - 0xC3	0x4	**VMECSR_ADER0_3**	Address Space Relocation Register VME A32	MSB
0xC4 - 0xC7	0x4	**VMECSR_ADER0_2**	Address Space Relocation Register VME A32	
0xC8 - 0xCC	0x4	**VMECSR_ADER0_1**	Address Space Relocation Register VME A32	
0xCC - 0xCF	0x4	**VMECSR_ADER0_0**	Address Space Relocation Register VME A32	LSB
0xD0 - 0xD3	0x4	**Reserved**	Not used	
0xD4 - 0xD7	0x4	**VMECSR_BCR**	VME64x CR/CSR Bit Clear Register	VME64x main Control & Status Control
0xD8 - 0xDC	0x4	**VMECSR_BSR**	VME64x CR/CSR Bit Set Register	
0xDC - 0xDF	0x4	**VMECSR_BAR**	VME64x CR/CSR Base Address Register	
0xE0 - 0xE3	0x4	**PVME_ADDERR**	VME64x Address Error Register	
0xE4 - 0xE7	0x4	**PVME_STAERR**	VME64x Status Error Register	
0xE8 - 0xEB	0x4	**PVME_RMW_MODE**	VME RMW Mode and Trigger	Programmed RMW
0xEC - 0xEF	0x4	**PVME_RMW_ADD**	VME RMW Address Pointer	
0xF0 - 0xF3	0x4	**PVME_RMW_DATCMP**	VME RMW Data Compare	
0xF4 - 0xF7	0x4	**PVME_RMW_DATUPT**	VME RMW Data Update	
0xF8 - 0xFB	0x4	**Reserved**	Not used	(Only IPV_1102)
0xFC - 0xFF	0x4	**Reserved**	Not used	

SEL Multiplexed =10→ DMA #1 (READ + WRITE Engines)

0xC0 - 0xC3	0x4	**SMEM_DDR2_CSR**	Four port DDR2 Controller Control & Status	
0xC4 - 0xC7	0x4	**IDMA_GCSR**	IDMA Global CSR	
0xC8 - 0xCC	0x4	**IDMA_RD_1_PCSR**	IDMA Pipeline Control Register Read Engine #0	
0xCC - 0xCF	0x4	**IDMA_WR_1_PCSR**	IDMA Pipeline Control Register Write Engine #0	
0xD0 - 0xD3	0x4	**IDMA_RD_1_CSR**	IDMA Read Engine #0 Control&Status	
0xD4 - 0xD7	0x4	**IDMA_RD_1_NDES**	IDMA Read Engine #0 Next DMA_Descriptor Pointer	
0xD8 - 0xDC	0x4	**IDMA_RD_1_CDES**	IDMA Read Engine #0 Current DMA_Descriptor Pointer	
0xDC - 0xDF	0x4	**IDMA_RD_1_WCNT**	IDMA Read Engine #0 WCNT and DMA Operation Status	
0xE0 - 0xE3	0x4	**IDMA_WR_1_CSR**	IDMA Write Engine #0 Control&Status	
0xE4 - 0xE7	0x4	**IDMA_WR_1_NDES**	IDMA Write Engine #0 Next DMA_Descriptor Pointer	
0xE8 - 0xEC	0x4	**IDMA_WR_1_CDES**	IDMA Write Engine #0 Current DMA_Descriptor Pointer	
0xEC - 0xEF	0x4	**IDMA_WR_1_WCNT**	IDMA Write Engine #0 WCNT and DMA Operation Status	
0xF0 - 0xFF	0x10	**Reserved**	Not used	

SEL Multiplexed = 11→ USER Agent_SW

0xC0 - 0xC3	0x4	**GPIO_A_OUT**	GPIO Group_A OUTPUT Register	
0xC4 - 0xC7	0x4	**GPIO_A_ENA**	GPIO Group_A TRI-STATE Register	
0xC8 - 0xCB	0x4	**GPIO_A_IN**	GPIO Group_A INPUT Register	
0xCC - 0xCF	0x4	**GPIO_A_POL**	GPIO Group_A POLARITY Register	Cassidian specific
0xD0 - 0xD3	0x4	**GPIO_B_OUT**	GPIO Group_B OUTPUT Register	
0xD4 - 0xD7	0x4	**GPIO_B_ENA**	GPIO Group_B TRI-STATE Register	
0xD8 - 0xDB	0x4	**GPIO_B_IN**	GPIO Group_B INPUT Register	
0xDC - 0xDF	0x4	**GPIO_A_POL**	GPIO Group_B POLARITY Register	Cassidian specific
0xE0 - 0xE3	0x4	**Msg_FIFO_CTL_0**	Message FIFO#0 Control & Status	Cassidian specific
0xE4 - 0xE7	0x4	**Msg_FIFO_CTL_1**	Message FIFO#1 Control & Status	Cassidian specific
0xE8 - 0xEB	0x4	**Msg_FIFO_CTL_2**	Message FIFO#2 Control & Status	Cassidian specific
0xEC - 0xEF	0x4	**Msg_FIFO_CTL_3**	Message FIFO#3 Control & Status	Cassidian specific
0xF0 - 0xF3	0x4	**Msg_FIFO_Port_0**	Message FIFO#0 Control & Status IO Port	Cassidian specific
0xF4 - 0xF7	0x4	**Msg_FIFO_Port_1**	Message FIFO#1 Control & Status IO Port	Cassidian specific
0xF8 - 0xFB	0x4	**Msg_FIFO_Port_2**	Message FIFO#2 Control & Status IO Port	Cassidian specific
0xFC - 0xFF	0x4	**Msg_FIFO_Port_3**	Message FIFO#3 Control & Status IO Port	Cassidian specific
0xF0 - 0xFF	0x10	**Reserved**	Not implemented	