

PCT-8424/8425/8426

Programmer's Guide

Table of Contents

1. General information

- 1.1 Introduction
- 1.2 Standard height and low-profile card design
- 1.3 Firmware version
- 1.4 Where to get more information, technical support

2. PCI Express controller

- 2.1 Introduction
- 2.2 PCI configuration register space
- 2.3 Register's mapping

3. Functional registers

- 3.1 Registers overview
- 3.2 Splitting the address space into blocks
- 3.3 Block of registers with 8-bit data (+0000 ÷ 03FC)
- 3.4 Block of DIO ports and edge detection circuits registers (+0400 ÷ 07FC)
- 3.5 Block of counters and timers registers (+0800 ÷ 0FFC)
- 3.6 Block of diagnostic registers (+3F00 ÷ 3FFC)
- 3.7 Comparison of PCT-8424/8425/8426 and PCD-8104/8105/8106 card's registers

4. Registers intended for digital inputs/outputs ports

- 4.1 Introduction
- 4.2 Digital port functionality
- 4.3 Registers DINReg0, ... , DINReg5 [RD]
- 4.4 Registers DINReg(2-0) and DINReg(5-3) [RD]
- 4.5 Registers DOUTReg0, ... , DOUTReg2 [WR]
- 4.6 Register DOUTReg(2-0) [WR]
- 4.7 Register DIOCfgReg [WR/RD]

5. Registers intended for DIO signal edge detection circuits

- 5.1 Introduction
- 5.2 Registers DINREReg(x-x) and DINFEReg(x-x) [WR]
- 5.3 Registers DINREStatusReg(x-x) and DINFEStatusReg(x-x) [RD]
- 5.4 Registers DINREClrReg(x-x) and DINFEClrReg(x-x) [WR]
- 5.5 Registers DINREIRQReg(x-x) and DINFEIRQReg(x-x) [WR]
- 5.6 Simplified schematic diagram of edge detection circuits register structure

6. Registers intended for interrupt handling circuits

- 6.1 Introduction
- 6.2 Interrupt handling circuit functionality
- 6.3 Register INTEnReg [WR]
- 6.4 Register IRQCfgReg [WR]
- 6.5 Register IRQStatusReg [RD]
- 6.6 Register IRQClrReg [WR]
- 6.7 Register TimerReg [WR/RD]
- 6.8 Simplified schematics diagram of interrupt handling circuits register structure

7. Registers intended for counters/timers circuits

- 7.1 Introduction
- 7.2 Counters and timers circuit functionality
- 7.3 Registers CNTxSetReg [WR]
- 7.4 Registers CNTxStrReg [RD]
- 7.5 Registers CNTxTim1Reg [RD]
- 7.6 Registers CNTxTim2Reg [RD]
- 7.7 Register CNTClrReg [WR]
- 7.8 Register CNTSetReg [WR]

7.9 Register CNTStrReg [WR]

7.10 Register CNTEnReg [WR]

8. Diagnostic registers (common to all card types)

8.1 Introduction

8.2 Register CardResetReg [WR]

8.3 Register CardResetStatusReg [RD]

8.4 Register CardSerNrReg [RD]

8.5 Register CardIDReg [RD]

8.6 Register FPGATypeReg [RD]

8.7 Register FPGAVerReg [RD]

9. Registers in address spaces BAR1 a BAR2

9.1 Introduction

9.2 Address space BAR1

9.3 Address space BAR2

1. General information

1.1 Introduction

This Programmer's Guide follows the PCT-8424/25/26 card User's Guide (hereinafter all types referred as PCT-842x) containing ...

- basic technical data,
- description of installation procedure
- and description of the connector pin assignment.

User's Guide is intended for a regular card user who only needs to install card and use it with already created programs.

Unlike the User's Guide, the Programmer's Guide contains ...

- description of the PCI Express controller built into the card,
- description of all functional registers of the card
- and description of register-level programming.

Programmer's Guide therefore enables programming using a system driver with an API offering direct access to the registers (in the case of Windows it is the `tedia_ep4gxa.dll` library), i.e. creating special programs or custom drivers (e.g. for various SCADA systems or for the Linux operating system).

1.2 Standard height and low-profile card design

DAQ PCI Express TEDIA cards are available in a standard height version (type designation PCT-842x) and low-profile version (type designation PCT-842x/LP). With the exception of the different locations of connectors and the applicable accessories, both version are identical and the information contained in this manual is therefore valid for both variants.

1.3 Firmware version

Current firmware version at the time this document was issued:

FPGA - firmware type:	26	(represented by a value 26 _H)
FPGA - firmware version:	0.A	(represented by a value 0A _H)

The FPGA firmware type is a control number assigned to the standard PCT-842x firmware. A different number represents either an incorrect firmware configuration (for example, intended for a different card) or custom firmware.

The FPGA firmware version is an additional parameter that defines the card properties.

Note: *The features described in this document reflect the specified firmware version. Later firmware versions (unless otherwise noted) will be backward compatible with the current version and will include improvements to existing functionality or brand new features.*

1.4 Where to get more information, technical support

Further useful information is available at...

website: <https://www.tedia.eu>

In doubt, you can contact the manufacturer's technical support:

address: TEDIA spol. s r. o., Zabelska 12, 31211 Plzen, Czech Republic

phone/e-mail: <https://www.tedia.eu/contacts>

tech. support: <https://www.tedia.eu/support>

Note: *Although this Programmer's Guide has been carefully reviewed, it can contain errors. If you suspect that some information is listed incorrectly, incompletely or inaccurately, please contact technical support.*

2. PCI Express controller

2.1 Introduction

All PCT-842x cards are equipped with a PCI Express bus controller core implemented in an FPGA gate array (i.e. the cards do not use any special PCI Express controller/bridge to the local bus interconnecting I/O peripherals).

The implementation of controller is single-functional (the card thus behaves as one PCI device) with three address spaces (BAR) mapped in the in 32-bit address MEM space.

Note: *Although the card's registers are mapped to the MEM space with 32-bit addressing, the DMA controller (if implemented) supports both 32-bit and 64-bit addressing modes.*

2.2 PCI configuration register space

The table below provides an overview of the selected registers from the PCI configuration register space.

address	register name	PCT-8424 PCT-8424/LP	PCT-8425 PCT-8425/LP	PCT-8426 PCT-8426/LP
01 _H ÷00 _H	Vendor ID	1760 _H (i.e. TEDIA VID)		
03 _H ÷02 _H	Device ID	0830 _H	0831 _H	0832 _H
08 _H	Revision ID	01 _H		
0B _H ÷09 _H	Class Code	118000 _H (i.e. PCI class "other data acquisition controller")		
13 _H ÷10 _H	BAR0	functional card's registers (MEM, 16kB, assigned by BIOS)		
17 _H ÷14 _H	BAR1	service purpose registers (firmware update, calibration constants, etc.) (MEM, 16kB, assigned by BIOS)		
1B _H ÷18 _H	BAR2	registers intended for the kernel part of the operating system driver (MEM, 4kB, assigned by BIOS)		
1F _H ÷1C _H	BAR3	unused		
23 _H ÷20 _H	BAR4	unused		
27 _H ÷24 _H	BAR5	unused		
2D _H ÷2C _H	Subsystem Vendor ID	1760 _H (i.e. TEDIA VID)		
2F _H ÷2E _H	Subsystem ID	0001 _H		
3C _H	Interrupt Line	number of IRQ channel (assigned by BIOS)		
3D _H	Interrupt Pin	01 _H (INTA)		

What is the information provided by the PCI configuration registers described above ...

- Vendor ID and Device ID are intended for identification of the card type in the system (in case of uncertainty, Subsystem Vendor ID and Subsystem ID may also be used, or Class Code)
- BARx are designed to determine the allocated resources, i.e. the starting address of the register blocks
- The Interrupt Line is designed to determine the current connection of the card's INT signal to the logic IRQ interrupt channel (resp. to the MSI channel in case of this interrupt mode)

2.3 Register's mapping

The following paragraphs describe general information about register mapping.

Register's mapping only in MEM space and not in I/O space as with TEDIA DAQ PCI cards ...

Mapping in I/O space is obsolete and very restrictive (it allows to allocate a total of 255 blocks of 256 bytes size to all PCI devices in the computer) and finds meaningful use only in operating systems (resp. development tools) that do not allow simple 32-bit or 64-bit addressing of MEM space (eg. MS-DOS).

Registers in BAR0 space ...

This BAR space contains all user registers (i.e. registers accessing I/O peripherals of the card).

The following chapters are dedicated, with a few exceptions, exclusively on the description of these registers.

Registers in BAR1/BAR2 space ...

These BAR spaces contain service purpose registers and registers intended for kernel part of the operating system driver.

3. Functional registers

3.1 Registers overview

The tables in the following paragraphs provide an overview of the functional registers implemented in the current firmware version (see Chapter 1). All functional registers described in this chapter are mapped in BAR0 space.

Warning: All registers, unless explicitly stated otherwise (e.g. registers whose initial value can be defined via the EEPROM contents), have zero values after power-up start or reset.
However, when program starts, it can not rely on this state because the registers can be set to different values by the previous program; they can be set to defined state either by programming values or by using the CardResetReg register.

Note: If you are creating a program that supports multiple types of card, it is also recommended to use a document containing tables with comparison of register maps for different types of cards.

3.2 Splitting the address space into blocks

The table below provides an overview of splitting of the entire BAR0 address space into several blocks according to I/O peripherals; this structure is used by all types of TEDIA DAQ PCI Express card.

offset within BAR0	description of registers block
+0000 ÷ 03FC	registers with 8-bit data (intended for easier migration from DAQ PCI cards)
+0400 ÷ 07FC	registers with 32-bit data (block of DIO ports and edge detection circuits)
+0800 ÷ 0FFC	registers with 32-bit data (block of counters and timers)
+1000 ÷ 3EFC	registers with 32-bit data (PCT-842x does not use this block)
+3F00 ÷ 3FFC	diagnostic registers (common to all types of cards)

3.3 Block of registers with 8-bit data (+0000 ÷ 03FC)

The table below provides a list of registers with 8-bit data.

The registers can be accessed with byte operand to the address specified in the table, or with dword operand, with valid data being transmitted on the lowest eight bits (higher bits are ignored during writing and zeroed during reading).

The program can only access addresses aligned to the dword (i.e. in integer multiple of 4) and it is not recommended to access addresses other than listed in the table.

offset within BAR0	description of register (write)	description of register (read)
+0000	DOUTReg0	DINReg0
+0004	DOUTReg1	DINReg1
+0008	DOUTReg2	DINReg2
+000C	(reserved for DOUTReg3)	DINReg3
+0010	(reserved for DOUTReg4)	DINReg4
+0014	(reserved for DOUTReg5)	DINReg5
+0080	DIOCfgReg	(read back the register value)
+0200	IRQCfgReg	IRQStatusReg
+0204	IRQClrReg	- - -
+0208	TimerReg	TimerReg
+020C	INTEnReg	(read back the register value)
+03F4		CardIDReg
+03F8	- - -	FPGATypeReg
+03FC	- - -	FPGAVerReg

Note: The mapping of the DIO port registers is identical to the DAQ PCIe cards PCD-8104/05/06.
The register mapping of the first three DIO ports is identical for all TEDIA DAQ PCIe cards.

3.4 Block of DIO ports and edge detection circuits registers (+0400 ÷ 07FC)

The table below provides a list of registers dedicated for access to DIO ports and signal rising/falling edge detection circuits with the possibility of triggering an interrupt.

The program can only access registers with dword operand at addresses aligned to the dword (i.e. in integer multiple of 4) and it is not recommended to access addresses other than listed in the table.

offset within BAR0	description of register (write)	description of register (read)
+0400	DOUTReg(2-0)	DINReg(2-0)
+0404	(reserved for DOUTReg(5-3))	DINReg(5-3)
+0410	DINREReg(2-0)	DINREStatusReg(2-0)
+0414	DINREClrReg(2-0)	- - -
+0418	DINFEReg(2-0)	DINFESTatusReg(2-0)
+041C	DINFEClearReg(2-0)	- - -
+0420	DINREReg(5-3)	DINREStatusReg(5-3)
+0424	DINREClrReg(5-3)	- - -
+0428	DINFEReg(5-3)	DINFESTatusReg(5-3)
+042C	DINFEClearReg(5-3)	- - -
+0440	DINREIRQReg(2-0)	(read back the register value)
+0444	DINFEIRQReg(2-0)	(read back the register value)
+0448	DINREIRQReg(5-3)	(read back the register value)
+044C	DINFEIRQReg(5-3)	(read back the register value)

Note: The function of digital ports, including mapping of the DIO port registers, is identical to the DAQ PCIe cards PCD-8104/05/06.

3.5 Block of counters and timers registers (+0800 ÷ 0FFC)

The table below provides a list of registers dedicated for access to registers of counters and timers.

The program can only access registers with dword operand at addresses aligned to the dword (i.e. in integer multiple of 4) and it is not recommended to access addresses other than listed in the table.

The register block on addresses +0800 ÷ 081C contains the registers of the counter and related timers (i.e. CNT0, CNT0Tim1 and CNT0Tim2). The analogous structure of the registers for counters/timers CNT1 - CNT23 is mapped to following addresses and these register blocks have exactly the same functionality as the registers block of CNT0.

offset within BAR0	description of register (write)	description of register (read)
+0800	CNT0SetReg	CNT0StrReg
+0804	(reserved)	(reserved)
+0808	(reserved)	CNT0Tim1Reg
+080C	(reserved)	CNT0Tim2Reg
+0810 ÷ 081C	(reserved)	(reserved)
+0820 ÷ 083C	registers of CNT1 counter and related timers (structure analogous to CNT0)	
+0840 ÷ 085C	registers of CNT2 counter and related timers (structure analogous to CNT0)	
+0860 ÷ 087C	registers of CNT3 counter and related timers (structure analogous to CNT0)	
+0880 ÷ 089C	registers of CNT4 counter and related timers (structure analogous to CNT0)	
+08A0 ÷ 08BC	registers of CNT5 counter and related timers (structure analogous to CNT0)	
+08C0 ÷ 08DC	registers of CNT6 counter and related timers (structure analogous to CNT0)	
+08E0 ÷ 08FC	registers of CNT7 counter and related timers (structure analogous to CNT0)	
+0900 ÷ 091C	registers of CNT8 counter and related timers (structure analogous to CNT0)	
+0920 ÷ 093C	registers of CNT9 counter and related timers (structure analogous to CNT0)	
+0940 ÷ 095C	registers of CNT10 counter and related timers (structure analogous to CNT0)	
+0960 ÷ 097C	registers of CNT11 counter and related timers (structure analogous to CNT0)	
+0980 ÷ 099C	registers of CNT12 counter and related timers (structure analogous to CNT0)	
+09A0 ÷ 09BC	registers of CNT13 counter and related timers (structure analogous to CNT0)	
continued on the next page...		

...continued from the previous page		
offset within BAR0	description of register (write)	description of register (read)
+09C0 ÷ 09DC	registers of CNT14 counter and related timers (structure analogous to CNT0)	
+09E0 ÷ 09FC	registers of CNT15 counter and related timers (structure analogous to CNT0)	
+0A00 ÷ 0A1C	registers of CNT16 counter and related timers (structure analogous to CNT0)	
+0A20 ÷ 0A3C	registers of CNT17 counter and related timers (structure analogous to CNT0)	
+0A40 ÷ 0A5C	registers of CNT18 counter and related timers (structure analogous to CNT0)	
+0A60 ÷ 0A7C	registers of CNT19 counter and related timers (structure analogous to CNT0)	
+0A80 ÷ 0A9C	registers of CNT20 counter and related timers (structure analogous to CNT0)	
+0AA0 ÷ 0ABC	registers of CNT21 counter and related timers (structure analogous to CNT0)	
+0AC0 ÷ 0ADC	registers of CNT22 counter and related timers (structure analogous to CNT0)	
+0AE0 ÷ 0AFC	registers of CNT23 counter and related timers (structure analogous to CNT0)	
+0B00	CNTClrReg	- - -
+0B04	CNTSetReg	- - -
+0B08	CNTStrReg	- - -
+0B0C	CNTEnReg	(read back the register value)

3.6 Block of diagnostic registers (+3F00 ÷ 3FFC)

The table below provides a list of registers dedicated for access to diagnostic and identification functions.

The program can only access registers with dword operand at addresses aligned to the dword (i.e. in integer multiple of 4) and it is not recommended to access addresses other than listed in the table.

offset within BAR0	description of register (write)	description of register (read)
+3FE0	CardResetReg	CardResetStatusReg
+3FF0	- - -	CardIDReg
+3FF4	- - -	CardSerNrReg
+3FF8	- - -	FPGATypeReg
+3FFC	- - -	FPGAVerReg

3.7 Comparison of PCT-8424/8425/8426 and PCD-8104/8105/8106 card's registers

The register structure of PCT-842x cards is an extension of the structure of PCD-810x cards and the functionality of all registers with the exception of counter/timer registers (address block +0800 ÷ 0FFC) is therefore identical.

4. Registers intended for digital inputs/outputs ports

4.1 Introduction

The following sections describe the registers related to digital inputs and outputs (see the overview in Chapter 3).

The DIO registers can be divided into a group of data registers...

DINReg0, ...5	six 8-bit digital port input registers (sets of signals DIO00÷07, DIO08÷15, ... , DIO40÷47)
DINReg(2-0)	32-bit digital port input register (combines DINReg0, DINReg1 and DINReg2)
DINReg(5-3)	32-bit digital port input register (combines DINReg3, DINReg4 and DINReg5)
DOUTReg0, ...2	three 8-bit digital port output registers (sets of signals DIO00÷07, DIO08÷15 and DIO16÷23)
DOUTReg(2-0)	32-bit digital port output register (combines DOUTReg0, DOUTReg1 and DOUTReg2)

and configuration register...

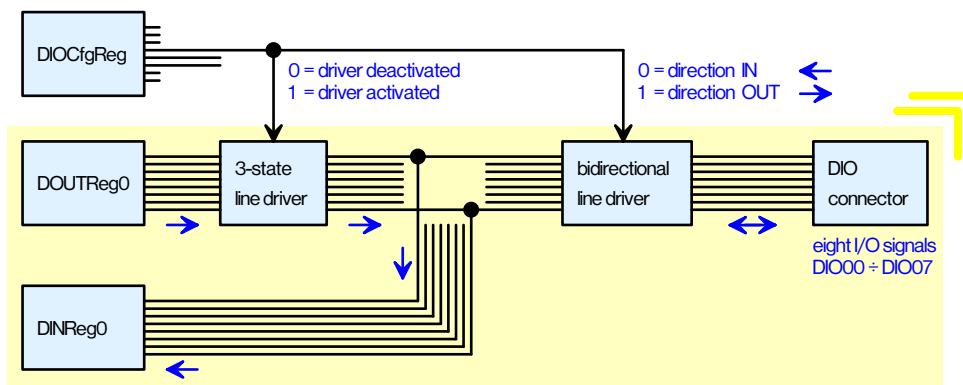
DIOCfgReg	register for configuration of direction DIO ports (i.e. selection of input or output port)
-----------	--

Note: Initial value (e.g. after power-up or soft-reset using the CardResetReg register) of all registers mentioned above can be defined via the EEPROM contents. Because the initial values stored in the EEPROM can be modified by the configuration program, the user can define the DIO ports status programmed immediately after turning on the computer without delay until the program starts.

4.2 Digital port functionality

The first three ports are designed as bidirectional, i.e. each port (= eight digital signals) can be individually set as input or output. The current state of the port can be obtained by reading the DINx register; in the case of configuration as an input port, the status of input signals is read, in the case of configuration as an output port, current data written to the output register are read back.

Further details can be seen in the picture below (only 8-bit port DIO0 is drawn, the part with the yellow background is included in the card three times, i.e. separately for DIO0, DIO1 and DIO2 port).



The next three digital ports (i.e. ports DIO3, DIO4 and DIO5) are configured as permanently inputs and simultaneously work as counter/timer inputs.

Location of ports on the card connectors

The first three ports (i.e. DIO0, DIO1 and DIO2) are connected to the KX1 - KX3 connectors located on the back edge of the card (they are accessible via adapter cable), the next three ports (i.e. DIO0, DIO1 and DIO2) are connected to the D-Sub 25 connector located on the card bracket.

Note: Although the second group of ports is referred as ports DIO3, DIO4 and DIO5, i.e. signals DIO24 to DIO47 (unified description of the DIO port registers for all types of cards), the User's Guide uses the more user-friendly designation DIN0, DIN1 and DIN2, i.e. signals DIN00 to DIN23.

Note: The mapping of the DIO port registers of PCT-842x cards is identical to the DAQ PCIe cards PCD-8104/05/06. The register mapping of the first three DIO ports is identical for all TEDIA DAQ PCIe cards.

4.3 Registers DINReg0, ... , DINReg5 [RD]

These registers are used to read the status of the digital port, each bit of the register accesses one signal of the 8-bit digital port (bits D0 of these registers access the signals DIO00/08/... /40; bits D7 access DIO07/15/... /47).

In the case the port is configured as an input port, the register provides a current status of input signals; in the case the port is configured as an output port, current data written to the output register are read back.

4.4 Registers DINReg(2-0) and DINReg(5-3) [RD]

These registers are an alternative to the 8-bit registers described in the previous paragraph and they are used to read the status of three digital ports at once (functionality remains identical to 8-bit registers).

The DINReg(2-0) register combines DINReg0, DINReg1 and DINReg2 (bits D00÷D23 provide a current value of ports DIO00÷DIO23, the highest eight bits are permanently zeroed).

The DINReg(5-3) register combines DINReg3, DINReg4 and DINReg5 (bits D00÷D23 provide a current value of ports DIO24÷DIO47, the highest eight bits are permanently zeroed).

4.5 Registers DOUTReg0, ... , DOUTReg2 [WR]

These registers are used to control the status of the output digital port, each bit of the register defines one signal of the 8-bit digital port (bits D0 of the registers control the signals DIO00/08/16; bits D7 control DIO07/15/23).

In the case the port is configured as an input port, data can be written to the DOUTReg register, but its contents do not affect the state of the signals. In the case the port is configured as an output port, the current contents of this register defines the status of the output signals.

Writing to these registers is allowed even if the port is set as input, their contents will be transferred to the signals immediately after the port is switched to output mode.

4.6 Register DOUTReg(2-0) [WR]

This register is an alternative to the 8-bit registers described in the previous paragraph and it is used to control the contents of three digital ports at once (functionality remains identical to 8-bit registers).

The DOUTReg(2-0) register combines DOUTReg0, DOUTReg1 and DOUTReg2 (bits D00÷D23 define the status of the output signal DIO00÷DIO23, the highest eight bits are ignored).

4.7 Register DIOCfgReg [WR/RD]

This register is used to configure bidirectional DIO ports as input or output.

D7	D6	D5	D4	D3	D2	D1	D0
RSRV					DIR2	DIR1	DIR0

DIR0	direction control of the DIO0 port
0	the outputs of the DOUT register are deactivated (i.e. the port works as an input)
1	the outputs of the DOUT register are activated (i.e. the port works as an output)
DIR1	direction control of the DIO1 port
0	the outputs of the DOUT register are deactivated (i.e. the port works as an input)
1	the outputs of the DOUT register are activated (i.e. the port works as an output)
DIR2	direction control of the DIO2 port
0	the outputs of the DOUT register are deactivated (i.e. the port works as an input)
1	the outputs of the DOUT register are activated (i.e. the port works as an output)
RSRV	reserved (to ensure forward compatibility it is recommended to write 0 and ignore readed value)

5. Registers intended for DIO signal edge detection circuits

5.1 Introduction

The following sections describe the registers related to signal edge detection circuits (see the overview in Chapter 3).

List of registers:

DINREReg(2-0)	enable of rising edge detection on DIO00÷DIO23 port signals
DINREStatusReg(2-0)	rising edge event flags on DIO00÷DIO23 port signals
DINREClrReg(2-0)	clear rising edge event flags on DIO00÷DIO23 port signals
DINFEReg(2-0)	enable of falling edge detection on DIO00÷DIO23 port signals
DINFESStatusReg(2-0)	falling edge event flags on DIO00÷DIO23 port signals
DINFEClearReg(2-0)	clear falling edge event flags on DIO00÷DIO23 port signals
DINREReg(5-3)	enable of rising edge detection on DIO24÷DIO47 port signals
DINREStatusReg(5-3)	rising edge event flags on DIO24÷DIO47 port signals
DINREClrReg(5-3)	clear rising edge event flags on DIO24÷DIO47 port signals
DINFEReg(5-3)	enable of falling edge detection on DIO24÷DIO47 port signals
DINFESStatusReg(5-3)	falling edge event flags on DIO24÷DIO47 port signals
DINFEClearReg(5-3)	clear falling edge event flags on DIO24÷DIO47 port signals
DINREIRQReg(2-0)	enable interrupts from edge detection circuits (rising edge detection on DIO00÷DIO23)
DINFEIRQReg(2-0)	enable interrupts from edge detection circuits (falling edge detection on DIO00÷DIO23)
DINREIRQReg(5-3)	enable interrupts from edge detection circuits (rising edge detection on DIO24÷DIO47)
DINFEIRQReg(5-3)	enable interrupts from edge detection circuits (falling edge detection on DIO24÷DIO47)

5.2 Registers DINREReg(x-x) and DINFEReg(x-x) [WR]

These registers are dedicated to enable rising edge detection (DINREReg), resp. falling edge detection (DINFEReg) on DIO port signals.

These registers have significant 24 lowest bits (bits D0 with value 1 enable edge detection on signals DIO00/DIO24; bits D23 with value 1 enable edge detection on signals DIO23/DIO47), the highest eight bits are ignored (value 0 recommended to ensure forward compatibility).

5.3 Registers DINREStatusReg(x-x) and DINFESStatusReg(x-x) [RD]

These registers are dedicated to determine the status of the flags of the edge detection circuits enabled by the DINREReg(x-x) and DINFEReg(x-x) registers.

These registers have significant 24 lowest bits (value 1 on bits D0 indicate active flags related to signals DIO00/DIO24; value 1 on bits D23 indicate active flags related to signals DIO23/DIO47), the highest eight bits are permanently zeroed.

5.4 Registers DINREClrReg(x-x) and DINFEClearReg(x-x) [WR]

These registers are dedicated to clearing selected flags of the edge detection circuits.

These registers have significant 24 lowest bits (bits D0 with value 1 clear flags related to signals DIO00/DIO24, bits D23 with value 1 clear flags related to signals DIO23/DIO47), the highest eight bits are ignored (value 0 recommended to ensure forward compatibility).

Writing a value of 1 to a defined bit generates a short pulse, so following write of 0 is not required. Clearing any combination of flags in a single write cycle is allowed.

5.5 Registers DINREIRQReg(x-x) and DINFEIRQReg(x-x) [WR]

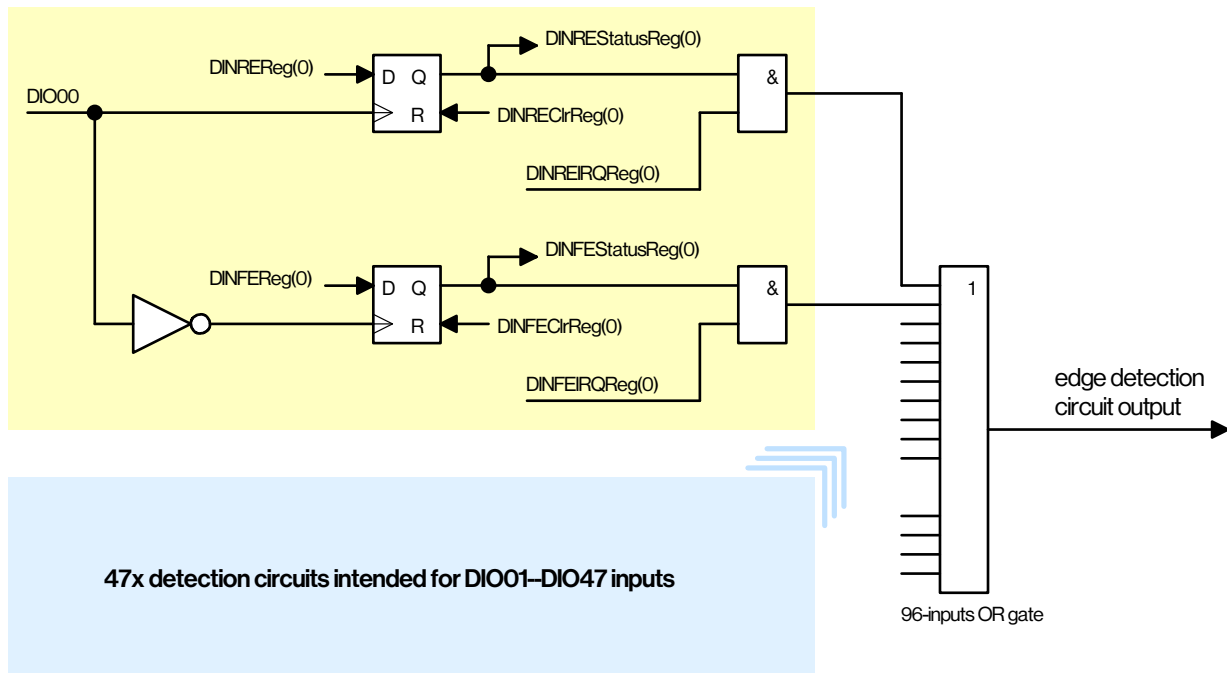
These registers are dedicated to enable system interrupt event generation related to edge detection circuit flags.

These registers have significant 24 lowest bits (bits D0 with value 1 enable interrupt event generation related to signals DIO00/DIO24; bits D23 with value 1 enable interrupt event generation related to signals DIO23/DIO47), the highest eight bits are ignored (value 0 recommended to ensure forward compatibility).

More detailed information can be found in the picture and description given in the following paragraph, resp. in a separate chapter with a description of interrupt circuits.

5.6 Simplified schematic diagram of edge detection circuits register structure

The figure below shows a simplified schematic diagram of the edge detection circuits, the connection with interrupt circuits is described in a separate chapter.



Each DIOxx signal is equipped with identical circuits that allow the independent detection of rising edge, falling edge or both edges on this signal.

The part of the schematics diagram highlighted in yellow shows the circuits dedicated for processing the signal DIO00, the blue marked part shows the identical circuits for the signals DIO01 to DIO47.

All 96 flags (available for reading via the DINREStatus and DINFStatus registers) allow to trigger an interrupt. Flags are processed by AND gates in the first step (DINREIRQReg and DINFEIRQReg with value 1 allow flags signal to pass through AND gate) and in the second step by OR gate.

As can be seen from the schematics diagram of the interrupt circuits (see the description in a separate chapter), the interrupt of the system is triggered by the first detected edge, i.e. by transitioning the OR gate output from 0 to 1. To further trigger an interrupt, it is therefore necessary for the interrupt program handler to process all interrupt events and then clear all the flags of the edge detection circuits.

6. Registers intended for interrupt handling circuits

6.1 Introduction

The following sections describe the registers related to interrupt handling circuits (see the overview in Chapter 3).

List of registers:

INTEnReg	interconnection of interrupt detection circuits (all registers described in this chapter) with card interrupt generation circuits (both INTA or MSI mode)
IRQCfgReg	enabling of general interrupt sources
IRQStatusReg	statuses of general interrupt sources
IRQClrReg	clearing of general interrupt flags
TimerReg	timestamp generator for periodically triggering interrupts

6.2 Interrupt handling circuit functionality

Interrupt handling circuits have the capability to cause a system interrupt event by one of the sources, or by a selected combination of interrupt sources. The card uses the following interrupt sources:

Timestamp generator

Allows to trigger an interrupt with a selected time period in the range of 1÷254 ms.

Digital inputs - simple mode compatible with DAQ PCI cards

Allows to trigger an interrupt with falling edge of selected DIO port signals.

Digital inputs - edge detection circuits used by DAQ PCI Express cards

Allows to trigger an interrupt by any combination of rising and falling edges on all DIO port signals.

Note: *For proper function of the interrupt circuits (see the schematics diagram on the following pages) it is necessary to consider, that interrupt of the system is triggered by the first event. To further trigger an interrupt, it is therefore necessary for the interrupt program handler to process all interrupt events and then clear all the flags (optional flags of the edge detection circuits).*

6.3 Register INTEnReg [WR]

These registers are dedicated to interconnection of interrupt detection circuits (all registers described in this chapter) with card interrupt generation circuits (both INTA or MSI mode).

D7	D6	D5	D4	D3	D2	D1	D0
INTEN	RSRV						

INTEN	activation of INTA/MSI control circuits
0	capture register generating control signal INTA or generating MSI is permanently zeroed
1	the capture register function is activated, i.e. the card may triggered a system interrupt
RSRV	reserved (to ensure forward compatibility it is recommended to write 0)

6.4 Register IRQCfgReg [WR]

These registers are dedicated to enable of general interrupt sources.

D7	D6	D5	D4	D3	D2	D1	D0
RSRV	DIN-X	RSRV	TIM	IRQ3	IRQ2	IRQ1	IRQ0

IRQ0	enables the trigger an interrupt derived from the falling edge of digital port DIO00 0 the capture register connected with digital input is disabled 1 the capture register functionality is enabled
IRQ1	enables the trigger an interrupt derived from the falling edge of digital port DIO08 0 the capture register connected with digital input is disabled 1 the capture register functionality is enabled
IRQ2	enables the trigger an interrupt derived from the falling edge of digital port DIO16 0 the capture register connected with digital input is disabled 1 the capture register functionality is enabled
IRQ3	enables the trigger an interrupt derived from the falling edge of digital port DIO24 0 the capture register connected with digital input is disabled 1 the capture register functionality is enabled
TIM	enables the trigger an interrupt derived from timestamp generator 0 the capture register connected with timestamp generator is disabled 1 the capture register functionality is enabled
DIN-X	enables the trigger an interrupt derived from signal edge detection circuits 0 the capture register connected with signal edge detection circuits 1 the capture register functionality is enabled
RSRV	reserved (to ensure forward compatibility it is recommended to write 0)

6.5 Register IRQStatusReg [RD]

This register is dedicated to determine the status of the capture registers enabled by the IRQCfgReg register.

D7	D6	D5	D4	D3	D2	D1	D0
RSRV	DIN-X	RSRV	TIM	IRQ3	IRQ2	IRQ1	IRQ0

IRQ0	the current status of the capture register connected with digital input DIO00 0 the register is not set, i.e. no falling edge has been detected since the last reset 1 the register is set, i.e. an interrupt request event has occurred since the last reset
IRQ1	the current status of the capture register connected with digital input DIO08 0 the register is not set, i.e. no falling edge has been detected since the last reset 1 the register is set, i.e. an interrupt request event has occurred since the last reset
IRQ2	the current status of the capture register connected with digital input DIO16 0 the register is not set, i.e. no falling edge has been detected since the last reset 1 the register is set, i.e. an interrupt request event has occurred since the last reset
IRQ3	the current status of the capture register connected with digital input DIO24 0 the register is not set, i.e. no falling edge has been detected since the last reset 1 the register is set, i.e. an interrupt request event has occurred since the last reset
TIM	the current status of the capture register connected with timestamp generator 0 the register is not set, i.e. no time stamp has been detected since the last reset 1 the register is set, i.e. an interrupt request event has occurred since the last reset
DIN-X	the current status of the capture register connected with edge detection circuits 0 the register is not set, i.e. no edge detection circuit event has been detected since the last reset 1 the register is set, i.e. an interrupt request event has occurred since the last reset
RSRV	reserved (to ensure forward compatibility it is recommended to ignore readed value)

6.6 Register IRQClrReg [WR]

This register is dedicated to clearing selected flags enabled by the IRQCfgReg register.

D7	D6	D5	D4	D3	D2	D1	D0
RSRV	DIN-X	RSRV	TIM	IRQ3	IRQ2	IRQ1	IRQ0

IRQ0	resets the capture register connected with digital input DIO00 0 idle write, the state of the capture register is not modified 1 the capture register is reset (subsequent writing 0 is not required)
IRQ1	resets the capture register connected with digital input DIO08 0 idle write, the state of the capture register is not modified 1 the capture register is reset (subsequent writing 0 is not required)
IRQ2	resets the capture register connected with digital input DIO16 0 idle write, the state of the capture register is not modified 1 the capture register is reset (subsequent writing 0 is not required)
IRQ3	resets the capture register connected with digital input DIO24 0 idle write, the state of the capture register is not modified 1 the capture register is reset (subsequent writing 0 is not required)
TIM	resets the capture register connected with timestamp generator 0 idle write, the state of the capture register is not modified 1 the capture register is reset (subsequent writing 0 is not required)
DIN-X	resets the capture register connected with edge detection circuits 0 idle write, the state of the capture register is not modified 1 the capture register is reset (subsequent writing 0 is not required)
RSRV	reserved (to ensure forward compatibility it is recommended to write 0)

6.7 Register TimerReg [WR/RD]

This register is dedicated to control a timestamp generator intended mainly to periodically trigger a system interrupt.

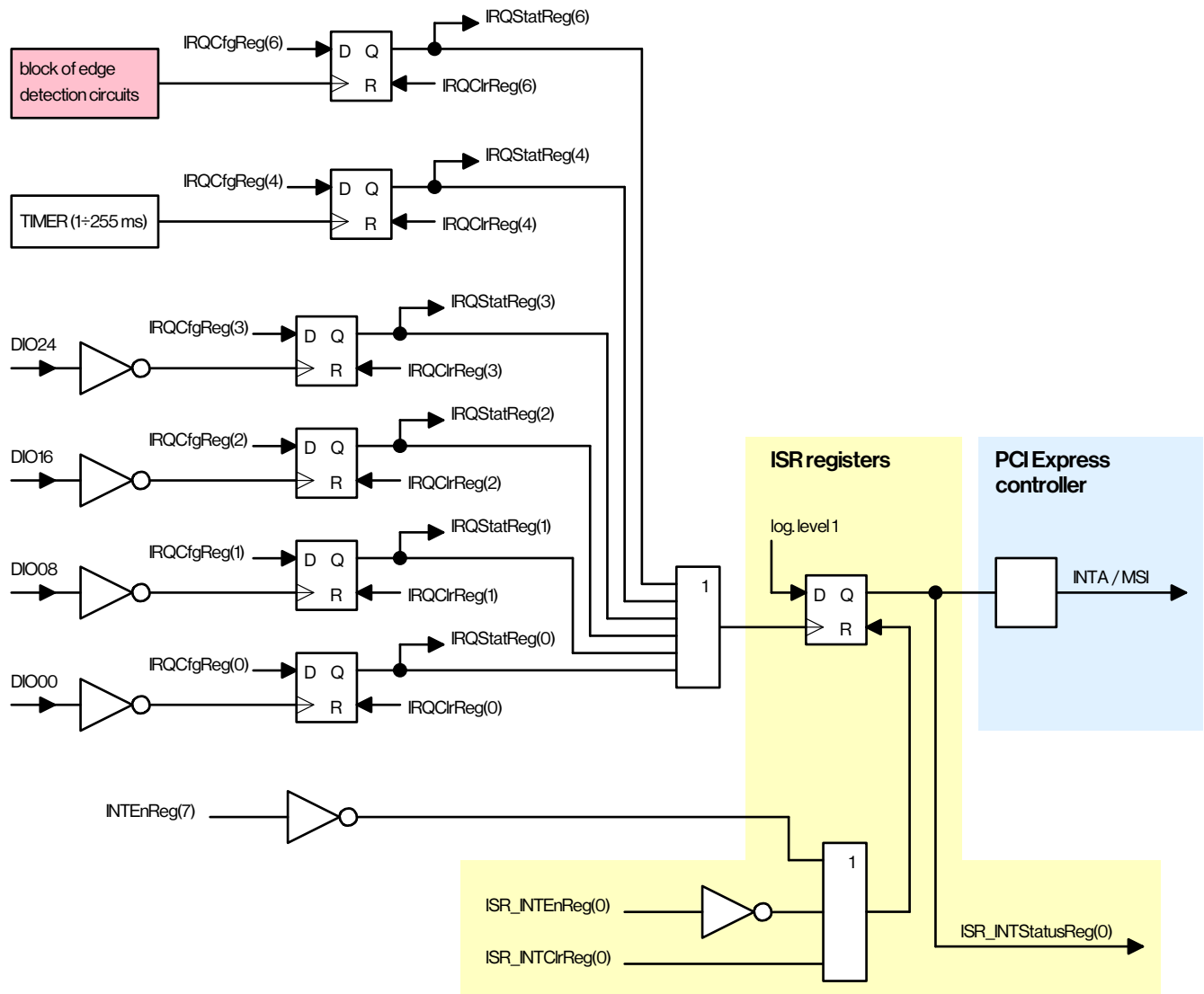
The initial value of the register is zero and the timestamp generator is stopped. By writing a non-zero value, the generator is started, the period is defined by the written value in milliseconds (up to 254 ms). By writing a zero value, the generator is stopped again.

The register is also important for reading (this register provides a current value of the timestamp counter incremented from zero every millisecond to the entered value reduced by one).

For example, by writing a value of 100, the first interrupt is triggered 100 ms after writing to the register and then every 100 ms. The values 0, 1, ..., 98, 99, 0, 1, ..., are provided by reading this register, the system interrupt is triggered at the moment of transition from 99 to 0.

6.8 Simplified schematics diagram of interrupt handling circuits register structure

The figure below shows a simplified schematic diagram of the interrupt handling circuits and interconnection with control/status registers; the edge detection circuits (highlighted in red) are described in a separate chapter.



Note: All three `ISR_***` registers are mapped in BAR2 space and their description is out of focus of this manual.

From a general point of view - the circuits highlighted with yellow background must be controlled within the ISR (i.e. in the OS kernel), the others can be controlled within the user program, within application driver (e.g. in the case of Windows, the DLL with an API containing abstract high-level functions) or also within the ISR.

The `ISR_INTEnReg(0)`, `ISR_INTCIrReg(0)` and `ISR_INTStatusReg(0)` signals are implemented identically on all TEDIA DAQ PCIe cards and allow to unify the ISR. The `ISR_INTEnReg(0)` signal is set to 1 after a power-up start or reset, therefore control via `INTEnReg(7)` can be also used to create a user specific kernel driver without the need for knowledge of mapping ISR registers in BAR2 space.

Internal solution of the standard `tedia_ep4gxa driver for Windows` fully complies with the information described above; the `ISR_INTEnReg(0)` and `ISR_INTCIrReg(0)` signals are controlled from the kernel part of this driver, as the only status information is used `ISR_INTStatusReg(0)`. The ISR is therefore completely independent of the type of card.

Note: Detailed information on interrupt handling register control is provided in the document "DAQ PCI/PCIe card's Windows system driver Programmer's Guide".

7. Registers intended for counters/timers circuits

7.1 Introduction

The following sections describe the registers related to counters/timers circuits (see the overview in Chapter 3).

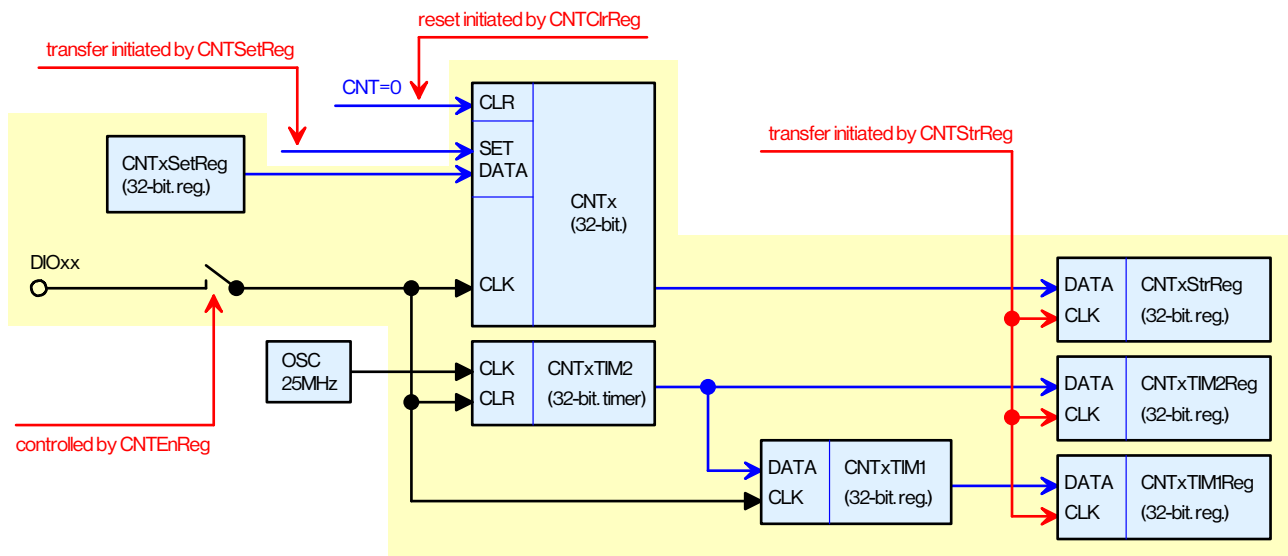
List of registers:

CNTxSetReg	24 registers intended for data written to the counters
CNTxStrReg	24 registers intended for data read from the counters
CNTxTim1Reg	24 registers intended for data read from the timers CNTxTIM2
CNTxTim2Reg	24 registers intended for data read from the capture registers CNTxTIM1
CNTClrReg	register intended for zeroing the value of selected counters
CNTSetReg	register intended for setting the value of selected counters
CNTStrReg	register intended for capture the value of selected counters to the registers
CNTEnReg	register intended for starting and stopping counters (i.e. enabling of input signal processing)

Note: The contents of all registers listed above are set to zero after power-up or reset (including the reset triggered by the CardResetReg register). The value of all CNTx counters and capture registers CNTxTIM1 is also reset, but the value of the CNTxTIM2 timers is set to the maximum value, i.e. $FFFFFFF_{16}$.

7.2 Counters and timers circuit functionality

The figure below shows a simplified schematic diagram of one block (highlighted with yellow background) containing main counter with related timer, capture register and a set of registers for reading and writing values. The PCT-842x card contains a total of 24 identical blocks.



While parts of function block (i.e. counter/timer/registers) and data paths are drawn in blue, operations that can be performed simultaneously for any selected subset of blocks are highlighted in red. The processed DIOxx signal (CNT0 processes signal DIO00, CNT23 processes signal DIO23) and reference time-base oscillator are drawn in black.

The CNTxSetReg registers contain values that can be transferred to CNTx counters by writing to the CNTSetReg register. Additionally, the counters can be reset by writing to the CNTClrReg register.

The CNTx counters process the DIOxx input signals depending on the state of the corresponding bit of the CNTEnReg register; this register is therefore used to stop or start the counter (the contents of the counter are then incremented by each pulse at the CLK input; depending on the type of PCT-842x, the active edge is the falling or rising edge).

The CNTxTIM2 timers incremented at 25MHz also process the DIOxx input signals depending on the CNTEnReg state, but unlike the counters, the timers by the active edge of these signals are reset and therefore provide information about the time elapsed since the last pulse. Unlike the counters, the timers also stop incrementing by reaching the limit value 4294967295, while the counters continue to increment again from the value 0.

The capture registers CNTxTIM1 also process the input signals DIOxx depending on the state of CNTEnReg; at the time of resetting the timers CNTxTIM2, the contents of these timers are transferred to the registers CNTxTIM1 and therefore provide information about the time elapsed between the last two pulses.

The values of counters CNTx with registers CNTxTIM1 and timers CNTxTIM2 are simultaneously transferred to the capture registers by writing to the CNTStrReg register, and then the contents of these capture registers can be read (at any time and in any order). The CNTxStrReg registers pass the counter value (i.e. the registered number of pulses), the CNTxTim1Reg registers pass the time elapsed between the last two registered pulses and the CNTxTim2Reg registers pass the time elapsed since the last registered pulse.

The timers are clocked at 25 MHz and pass a value in the range of a 32-bit number (i.e. 0 to about 171 seconds in 40 ns increments).

The transfer of values to counters, resetting (i.e. zeroing) counters, control start/stop and capture of current values of counters/timers/registers can be performed independently for one selected functional block, the selected subset of blocks or all blocks at once.

7.3 Registers CNTxSetReg [WR]

These 24 registers store values, which are transferred to counters by writing to the CNTSetReg register.

The transfer of values can be performed independently for any selected subset of counters/timers/registers blocks.

7.4 Registers CNTxStrReg [RD]

These 24 registers are used to capture the current values of the counters by writing to the CNTStrReg register.

The counters provide a value in the 32-bit range incremented by each pulse with the possibility of software resetting by the CNTClrReg register or setting any value by the CNTSetReg register. The counters work in overflowing mode, i.e. after the value 4294967295 continue to increment again from the value 0.

7.5 Registers CNTxTim1Reg [RD]

These 24 registers are used to capture the current values of the CNTxTim1 registers containing information about the time elapsed between the last two registered pulses by writing to the CNTStrReg register.

The data format is identical to the CNTxTim2Reg.

7.6 Registers CNTxTim2Reg [RD]

These 24 registers are used to capture the current values of the CNTxTim2 timers containing information about the time elapsed since the last registered pulse by writing to the CNTStrReg register

The timers are clocked at 25 MHz and pass a value in the range of a 32-bit number (i.e. 0 to about 171 seconds in 40 ns increments) and work in non-overflowing mode, i.e. the timers stop incrementing by reaching the limit value 4294967295 (i.e. this value indicates that 4294967295 x 40 ns or more has elapsed since the last pulse; if the CNTxTim1Reg value is also zero, it means that no pulse has been detected yet).

The timers, unlike other registers, are set to the maximum value after power-up or reset.

7.7 Register CNTClrReg [WR]

This register is used to reset (i.e. zeroed) any selected subset of counters. It does not affect the contents of the CNTxTim1 registers and CNTxTim2 timers.

The register have significant 24 lowest bits (bit D0 with value 1 clears counter CNT0, bit D23 with value 1 clears counter CNT23), the highest eight bits are ignored (value 0 recommended to ensure forward compatibility). Writing a value of 1 to a defined bit generates a short pulse, so following write of 0 is not required. Resetting of any combination of counters in a single write cycle is allowed.

7.8 Register CNTSetReg [WR]

This register is used to transfer of contents CNTxSetReg registers into any selected subset of counters. It does not affect the contents of the CNTxTim1 registers and CNTxTim2 timers.

The register have significant 24 lowest bits (bit D0 with value 1 sets value of counter CNT0, bit D23 with value 1 sets value of counter CNT23), the highest eight bits are ignored (value 0 recommended to ensure forward compatibility). Writing a value of 1 to a defined bit generates a short pulse, so following write of 0 is not required. Programming any combination of counters in a single write cycle is allowed.

7.9 Register CNTStrReg [WR]

This register is used to transfer contents of CNTx into CNTxStrReg registers, contents of CNTxTim1 into CNTxTim1Reg registers and contents of CNTxTim2 into CNTxTim2Reg registers (always a whole trio of transfers at once, the transfer cannot be performed separately); transfers can be performed for any selected subset of functional blocks.

The register have significant 24 lowest bits (bit D0 with value 1 performs the transfer within block 0, bit D23 with value 1 performs the transfer within block 23), the highest eight bits are ignored (value 0 recommended to ensure forward compatibility). Writing a value of 1 to a defined bit generates a short pulse, so following write of 0 is not required. Transfers within any combination of blocks in a single write cycle is allowed.

7.10 Register CNTEnReg [WR]

This register is used to enable or disable processing of DIOxx signals by counters/timers.

The register have significant 24 lowest bits (bit D0 with value 1 enable processing of DIO00 signal within block 0, bit D23 with value 1 enable processing of DIO23 signal within block 23), the highest eight bits are ignored (value 0 recommended to ensure forward compatibility).

8. Diagnostic registers (common to all card types)

8.1 Introduction

The following sections describe diagnostic registers (see the overview in Chapter 3).

List of registers:

CardResetReg	set all registers of the card to a defined state (identical to state after power-up)
CardResetStatusReg	status of running the procedure of setting all registers to the defined state
CardSerNrReg	provides a unique card serial number
CardIDReg	provides the status of the dual DIP switch (allows to differentiate up to 4 cards)
FPGATypeReg	provides the FPGA firmware type value
FPGAVerReg	provides the FPGA firmware version value

8.2 Register CardResetReg [WR]

Writing the value 5043384B_H to this register causes an immediate reset (i.e. set to zero, unless otherwise stated in the register description) all registers except all DOUTRegX registers and DIOCfgReg register.

Immediately after resetting, the preconfigured content is read from the on-board EEPROM and stored to the DOUTRegX and DIOCfgReg registers; this procedure typically takes 1 ms and its progress is signaled by the CardResetStatusReg register. All EEPROM values mentioned above can be modified by the free available configuration program.

Warning: During the ongoing register setup the program may only access the CardResetStatusReg register.

8.3 Register CardResetStatusReg [RD]

This register provides an access to flag indicating status of the procedure of setting all registers to the defined state triggered by writing to CardResetReg register.

The register has only the lowest bit significant (all others are permanently zero); a status bit with a value of 1 signals the ongoing setting of the card registers, a value of 0 then corresponds to the idle state.

8.4 Register CardSerNrReg [RD]

This register provides an access to the unique serial number of the card (32-bit integer format).

8.5 Register CardIDReg [RD]

This register provides an access to the status of the dual DIP switch (allows to differentiate up to 4 cards each other).

The register is mapped at two addresses, status is passed on the lowest two bits, the upper six bits (or 30 bits) are permanently zeroed.

8.6 Register FPGATypeReg [RD]

This register provides an access to the FPGA firmware type value in the range 0 to 255.

The register is mapped at two addresses, status is passed on the lowest eight bits, the upper 24 bits are permanently zeroed.

Note: The value of the standard card firmware type was mentioned in Chapter 1.

8.7 Register FPGAVerReg [RD]

This register provides an access to the FPGA firmware version value in the range 0 to 255.

The register is mapped at two addresses, status is passed on the lowest eight bits, the upper 24 bits are permanently zeroed.

Note: The value of the standard card firmware version was mentioned in Chapter 1.

9. Registers in address spaces BAR1 a BAR2

9.1 Introduction

While the previous chapters described, with a few exceptions, the functional registers in the address space BAR0, the following paragraphs will be dedicated to the registers in the address spaces BAR1 and BAR2.

Warning: *The registers mapped in the BAR1 and BAR2 address spaces are subject to change depending on the firmware version and, unlike the functional registers in BAR0, backward or forward compatibility is not guaranteed. Therefore, the software that uses these registers must modify its functions not only by the card type, but also by the content of the FPGATypeReg and FPGAVerReg registers.*

9.2 Address space BAR1

The address space BAR1 provides an access to service registers (interface for FPGA firmware update, calibration constants, configurable start-up settings of registers, ...) and their description is out of focus of this manual.

Note: *The full specification of BAR1 registers is subject to NDA and can be provided only in justified cases.*

9.3 Address space BAR2

The address space BAR1 provides an access to registers intended for kernel part of the operating system driver (ISR routine registers, DMA registers, ...) and their description is out of focus of this manual.

Note: *The full specification of BAR1 registers is subject to NDA and can be provided only in justified cases.*