

VME-CHIP of TCS-9U-card

H. Bergauer, K. Kastner, B. Neuherz, M. Padrta, T. Schreiner, A. Taurok



Feb-07

Version 0x1009

Table of contents

1	Introduction	3
2	VME chip of TCS-9U-card	3
2.1	Versionshistory	3
2.2	Hardware	3
2.3	Firmware	3
2.4	Features of the VME-CHIP-TCS (V1009) NEW	3
2.5	Fixed test-outputs	4
2.6	Programmable test-outputs	4
2.7	VME access	4
2.7.1	Base address	4
2.7.2	AM and datatransfer	4
2.8	Chip selection on TCS-9U-card	4
2.9	VME chip TCS register	4
2.9.1	VME chip TCS address-table	4
2.9.2	Register for Programmable-chips-configuration	5
2.9.2.1	CMD_ENPROG-register	5
2.9.2.2	CMD_NPROG-register	6
2.9.2.3	CMD_INIT-register	6
2.9.2.4	STAT_INIT-register	6
2.9.2.5	STAT_DONE-register	6
2.9.3	General registers	6
2.9.3.1	Command pulse register	6
2.9.3.2	Status pulse register	7
2.9.3.3	Configuration register TCS-chip	7
2.9.3.4	Configuration register TCSM-chip	7
2.9.4	Chip ID and version registers	7
2.9.4.1	Definitions	7
2.9.4.2	Settings	7
2.9.4.3	Chip ID and version registers addresses	7
2.10	DTACK/BERR-generation	8

1 Introduction

The VME-CHIP-TCS works with the VME64x chip TCS as controller for the VME-bus of the TCS-9U-card. There are VME-registers on it as the Registers for Programmable-chips-configuration, General registers and Chip ID / version registers. The VME-accesses to the TCS/TCSM-chip are made via this chip too.

2 VME chip of TCS-9U-card

2.1 Versionshistory

- **V1009: based on V1007**, but DTACK_EXT and BERR_EXT are used as positive active signals to VME64x-chip. (HB071106) **NEW**
- V1008: based on V1007, but NSYSRES generates NPROG_TCS and NPROG_TCSM (additional diodes necessary on board: JP13/1-2 and JP24/1-2). DONE_TCS and DONE_TCSM were delayed and used to enable NDTACK_TCS and NDTACK_TCSM (against "DTACK"-error after power-up) to work with TCS-chip V7 and TCSM-chip V2. (HB071106)
- V1007: based on V1006, but no NBERR-signals, because not used on TCS-board. (HB, 131006)
- V1006: new synchronous design. TCS/TCSM-chip access made with DSSYNC for read and write accesses, write/read for general-register and configuration-register implemented. Lines ASCYC, ASSYNC, ASPULS and D08_E from VME64x-chip represent the CARD_NR[3:0] – from jumpers S31-S28 on board. DSSYNC with two clock delays (ADDR_DEC_FDL_V1_0 and output-FF) used for read/write of TCS/TCSM-chip. DTACK_EXT and BERR_EXT are used as negative active signals to VME64x-chip (because at power-up configuration of VME64X-CHIP is faster than configuration of VME-CHIP and therefore wrong DTACK and BERR signals are generated after configuration, which causes LEDs="on" of CAEN-controller). (HB090506)
- V1005: similar to V1001, but new DTACK-generation implemented. (HB, 040705)
- V1004: like V1001, but new schematcs. (HB, 040705) – **Error in DTACK-generation. DO NOT USE!!!**
- V1003: not used!
- V1002: not used!
- V1001: single access and block transfer access – version ok. (HB, 180604)
- V1000: only single access

2.2 Hardware

The VME-FDL-chip is an Altera EP1K10TC100-3.

2.3 Firmware

chip_id: 0x00015n21 (n = CARD_NR from jumpers S31 - S28)
 version: 0x00001009

2.4 Features of the VME-CHIP-TCS (V1009) **NEW**

- VME-access to TCS- and TCSM-chip with DSSYNC for read and write (one clock delays).
- DTACK generation from TCS- and TCSM-chip.
- Register for commands and status (see 2.9.1 VME chip TCS address-table).

- Registers for chip_ID and version (cardnumber from VME64x-chip: jumpers S31-S28 on board).
- Configuration of TCS/TCSM-chip via VME (changes in hardware of mezz957 necessary!!!).
- Fixed test-outputs.

2.5 Fixed test-outputs

TST_CLK_VME: CLK_VME (40 MHz clock)

2.6 Programmable test-outputs

There are no programmable test-outputs on the VME-CHIP-TCS.

2.7 VME access

2.7.1 Base address

Base address of all GT-slaves is encoded on A31-A25 (A24 not used), because of address space of GTL-6U-card. See definition in VME64x-chip for TCS.

2.7.2 AM and datatransfer

AM=0x0D and 0x09 „extended data access“ - for single access.

AM=0x0F and 0x0B „extended block transfer“ - for block transfer access.

D16 „word access“ - for all accesses.

See definitions in VME64x-chip for TCS.

2.8 Chip selection on TCS-9U-card

With the VME addresses A23-A20 the chip selection is done on the TCS-9U-card.

A23	A22	A21	A20	Chip-name
0	0	0	0	VME chip TCS
0	1	0	0	TCS-chip
1	0	0	0	TCSM-chip

2.9 VME chip TCS register

A23	A22	A21	A20
0	0	0	0

2.9.1 VME chip TCS address-table

The address-table lists the address-offset which has to be combined with the base-address of the card.

A23-A00 => Register-name

Register for Programmable-chips-configuration:

0x000000 0x00000003 w/r => CMD_ENPROG-register (see 2.9.2.1)

0x000002 0x00000003 w/r => CMD_NPROG-register (see 2.9.2.2)

0x000004 0x00000003 w/r => CMD_INIT-register (see 2.9.2.3)

0x000006 0x00000003 r => STAT_INIT-register (see 2.9.2.4)

0x000008 0x00000003 r => STAT_DONE-register (see 2.9.2.5)

General registers:

0x00000A 0x000000F w => Command pulse register (see 2.9.3.1)
 0x00000C 0x000000F r => Status pulse register (see 2.9.3.2)
 0x000010 0x0000001 w => Configuration register TCS-chip (see 2.9.3.3)
 0x000012 0x0000001 w => Configuration register TCSM -chip (see 2.9.3.4)

Chip ID and version registers: (see 2.9.4)

0x000020 0x000000FF r => chip_id_register_3
 0x000022 0x000000FF r => chip_id_register_2
 0x000024 0x000000FF r => chip_id_register_1
 0x000026 0x000000FF r => chip_id_register_0
 0x000028 0x000000FF r => version_register_3
 0x00002A 0x000000FF r => version_register_2
 0x00002C 0x000000FF r => version_register_1
 0x00002E 0x000000FF r => version_register_0

Access to/from TCS-chip:

0x4XXXXX => see TCS-chip

Access to/from TCSM-chip:

0x8XXXXX => see TCSM-chip

2.9.2 Register for Programmable-chips-configuration**Register for Programmable-chips-configuration:**

A31..A24	A23..A20	A19..A06	A05..A01,(00)
8 bits		14bits	5bits
Base address	0000	XXXX	Registers

The TCS-chip and the TCSM-chip (Virtex-II) is configurable by configuration device and by VMEbus instructions. The selection is made by jumpers. The register-definition for configuration by VMEbus shall be a standard. See P:\Lab3Lib\Altera\Lab3_altera\sch\xilinx_conf.

<i>Register names</i>	D7..D2	D1	D0
CMD_ENPROG	-	ENPROG_TCSM	ENPROG_TCS
CMD_NPROG	-	NPROG_TCSM	NPROG_TCS
CMD_INIT	-	INIT_TCSM	INIT_TCS
STAT_INIT	-	INIT_TCSM	INIT_TCS
STAT_DONE	-	DONE_TCSM	DONE_TCS

2.9.2.1 CMD_ENPROG-register

0x000000 => CMD_ENPROG-register (write/read)

Bit 0 of the CMD_ENPROG-register allows sending the configuration bits via VME-bus to the TCS-chip.

Bit 1 of the CMD_ENPROG-register allows sending the configuration bits via VME-bus to the TCSM-chip.

Remark: Also the jumper EN_CONF_BY_VME has to be OFF (=1) to allow configuration via VME. The jumper is used for safety to prevent unintended configuration access via VME.

2.9.2.2 *CMD_NPROG-register*

0x000002 => CMD_NPROG-register (write/read)

Data-bit 0 = 1 of this register set the NPROG-signal of TCS-chip active. Then it should be reset to =0. Then the TCS-chip enters into the configuration procedure. The FPGA either waits for configuration data (slave mode) sent via VME or starts to read configuration bits from a serial PROM (master mode).

Data-bit 1 = 1 of this register set the NPROG-signal of TCSM-chip active. Then it should be reset to =0. Then the TCSM-chip enters into the configuration procedure. The FPGA either waits for configuration data (slave mode) sent via VME or starts to read configuration bits from a serial PROM (master mode).

2.9.2.3 *CMD_INIT-register*

0x000004 => CMD_INIT-register (write/read)

Data-bit 0 = 1 of this register set the NINIT-signal of TCS-chip active.

Data-bit 1 = 1 of this register set the NINIT-signal of TCSM-chip active.

2.9.2.4 *STAT_INIT-register*

0x000006 => STAT_INIT-register (read)

Read the status of the NINIT-signal of TCS-chip (data-bit 0)

Read the status of the NINIT-signal of TCSM-chip (data-bit 1)

2.9.2.5 *STAT_DONE-register*

0x000008 => STAT_DONE-register (read)

Read the status of the DONE-signal of TCS-chip (data-bit 0) and TCSM-chip (data-bit 1).

After a successful configuration the TCS-chip and TCSM-chip set DONE =1.

2.9.3 General registers

<i>Register names</i>	D7..D4	D3	D2	D1	D0
Command_Pulse_Reg	not used	PWRDWN_TCSM (pulse)	RESET_TCSM (pulse*)	PWRDWN_TCS (pulse)	RESET_TCS (pulse*)
Status_Pulse_Reg	not used	RUNNING	INACTIVE	CLK_LOCKED_TCSM	CLK_LOCKED_TCS

*) also generated by SYSRES

2.9.3.1 *Command pulse register*

0x00000A => Command-pulse-register (write)

D0: RESET_TCS = 1 sends a high active pulse to the TCS-chip for reset activities.

D1: PWRDWN_TCS = 1 sends a low active pulse to the TCS-chip setting it into power down mode. NPWRDWN_B is sent as an open drain signal from the VME-TCS-chip to the TCS-chip.

The power-down sequence enables a designer to set the device into a low-power, inactive state. The sequence is initiated by pulling the PWRDWN_B pin Low. To monitor power-down status, observe the PWRDWN_B pin. When asserted, power-down has completed. After a successful wake-up, the status pin de-asserts. While powered down, the only active pins are the PWRDWN_B and DONE. All inputs are off and all outputs are 3-stated. While in the POWERDOWN state, the Power On Reset (POR) circuit is still active, but it does not reset the device if V_{CCINT}, V_{CCO}, or V_{CCAUX} falls below its minimum value. The POR circuit waits until the PWRDWN_B pin is released before resetting the device. Also, the PROG_B pin is not sampled while the device is in the POWERDOWN state. The PROG_B pin becomes active when the PWRDWN_B pin

is released. Therefore, the device cannot be reset while in the POWERDOWN state. The wake-up sequence is the reverse of the power-down sequence.

D2: **RESET_TCSM = 1** sends a high active pulse to the TCS-chip for reset activities.

D3: **PWRDWN_TCSM = 1** sends a low active pulse to the TCSM-chip setting it into power down mode. NPWRDWN_B is sent as an open drain signal from the VME-TCS-chip to the TCSM-chip.

2.9.3.2 Status pulse register

0x00000C => Status-pulse-register (read)

D0: **CLK_LOCKED_TCS = 1**

D1: **CLK_LOCKED_TCSM = 1**

The DCM module of the TCS/TCSM chips are locked to the 40 MHz clock.

This status bit has to be checked immediately after the configuration of the TCS and TCSM chips and before any other actions. If the chips do not lock then either the clock signal from the TIM board or the on-board oscillator are bad.

D2: **INACTIVE = 1** all outputs from the TCS board are disabled. Push the RUN button to activate the board or send a SET_RUNNING command via VME.

D3: **RUNNING = 1** board is active.

If RUNNING = 0, send a SET_RUNNING command via VME.

2.9.3.3 Configuration register TCS-chip

0x000010 => Configuration register TCS-chip (write)

The register is used to load the configuration bits into the TCS-chip (Virtex-II).

A write access to this register generates a CCLK and sends the data-bit 0 as DIN-signal to the TCS-chip, if the CMD_ENPROG-register bit has been set before. The VME accesses are repeated until the last bit has been loaded into the TCS-chip.

2.9.3.4 Configuration register TCSM-chip

0x000012 => Configuration register TCSM-chip (write)

The register is used to load the configuration bits into the TCSM-chip (Virtex-II).

A write access to this register generates a CCLK and sends the data-bit 0 as DIN-signal to the TCSM-chip, if the CMD_ENPROG-register bit has been set before. The VME accesses are repeated until the last bit has been loaded into the TCSM-chip.

2.9.4 Chip ID and version registers

2.9.4.1 Definitions

Chip_id_register and version_register have fixed values in the hardware. These registers have read access only .

For definition of the chip_id-registers see P:\GlobalTrigger\Documentation\GT-6U-chip_id.pdf

The versions 0x00000000 - 0x00000FFF are used for tests.

The versions 0x00001000 - 0xFFFFFFFF are used for runs in CMS.

2.9.4.2 Settings

TCS-9U-card:

chip_id: 0x00015n21 (n = CARD_NR from jumpers S31 - S28)

version: 0x00001009

2.9.4.3 Chip ID and version registers addresses

Chip ID and version registers:			
A31..A24	A23..A20	A19..A06	A05..A01,(00)
8 bits		14bits	5bits
Base address	0000	XXXX	Registers

A23-A00 => Register-name

0x000020 => chip_id_register_3 (read)

D7	D6	D5	D4	D3	D2	D1	D0
chip_ID [31..24]							

0x000022 => chip_id_register_2 (read)

D7	D6	D5	D4	D3	D2	D1	D0
chip_ID [23..16]							

0x000024 => chip_id_register_1 (read)

D7	D6	D5	D4	D3	D2	D1	D0
chip_ID [15..08]							

0x000026 => chip_id_register_0 (read)

D7	D6	D5	D4	D3	D2	D1	D0
chip_ID [07..00]							

0x000028 => version_register_3 (read)

D7	D6	D5	D4	D3	D2	D1	D0
version [31..24]							

0x00002A => version_register_2 (read)

D7	D6	D5	D4	D3	D2	D1	D0
version [23..16]							

0x00002C => version_register_1 (read)

D7	D6	D5	D4	D3	D2	D1	D0
version [15..08]							

0x00002E => version_register_0 (read)

D7	D6	D5	D4	D3	D2	D1	D0
version [07..00]							

2.10 DTACK/BERR-generation

Writing to writeable registers and reading from readable registers generates a DTACK signal. Access to/from TCS/TCSM-chip generates a DTACK. No BERR signal generated.