FTM Firmware Specifications

Patrick Vogler[1], Quirin Weitzel

v5.1 - July 2012

[1]Contact for questions and suggestions concerning this document: `patrick.vogler@phys.ethz.ch`

Contents

Chapter 1

Introduction

The FTM (FACT Trigger Master) board collects the trigger primitives from all 40 FTU boards (FACT Trigger Unit) and generates the trigger signal for the FACT camera. The trigger logic is a 'n-out-of-40' majority coincidence of all trigger primitives. Beside the trigger, the FTM board also generates a trigger-ID (see chapter 2). It is controlled from outside via ethernet. Two auxiliary RS-485 interfaces are also available in hardware, but not yet supported in firmware.

In addition to the trigger, the FTM board also generates other fast control signals: Time-Marker (TIM), DRS [1] reference clock (CLD) and reset. These four fast control signals are distributed to the FAD (FACT Analog to Digital) boards via two FFC (FACT Fast Control) boards. The FTM board also provides, via the TIM line, the signal for the DRS timing calibration. In order to generate the CLD DRS reference clock, as well as the time-marker signal for DRS timing calibration, the FTM board uses a clock conditioner [2].

The FTM board has two time counters, the 'timestamp counter' and the 'on-time counter'. While the 'timestamp counter' runs continuously, the 'on-time counter' only counts when the camera trigger is enabled, i.e. when no FAD board is busy and not during the dead time. Both counters are reset at the beginning and at the end of a run.

The FTM board further serves as slow control master for the 40 FTU boards. The slow control of the FTU boards and the distribution of the trigger-ID to the FAD boards are performed via dedicated RS-485 buses. Because the FAD as well as the FTU boards are arranged in crates of 10 boards each, the FTM board has four connectors, one for each crate. Running over these connectors there are two RS-485 buses (one for FTU slow control and one for the trigger-ID) besides the busy signal from the FAD boards and the crate reset.

In addition, the FTM board controls the two FLPs (FACT Light Pulser) via four LVDS signals each. Light pulser 1 (also known as 'external lightpulser') is located in the mirror dish, light pulser 2 (also known as 'internal lightpulser') inside the camera shutter. There are also digital auxiliary in- and outputs according to the NIM (Nuclear Instrumentation Module) standard, for example for external triggers and veto, and to have the signals accessible.

The main component of the FTM board is a FPGA (Xilinx Spartan XC3SD3400A-4FGG676C), fulfilling the main functions within the board. The purpose of this document is to provide specifications needed for the understanding of the firmware of this FPGA and the software (called 'FTMcontrol' in the following) controlling the FTM board. For further information about the FTM board hardware please refer to [3].

Chapter 2

Trigger-ID

For each processed trigger the FTM board generates a unique trigger-ID to be broadcasted to all FAD boards and added to the event data. This trigger-ID consists of a 32 bit trigger number, a two byte trigger type indicator and a checksum. The transmission protocol for the trigger-ID broadcast is shown in table 2.1.

byte no	content
0	Trigger-No first byte (least significant byte)
1	Trigger-No second byte
2	Trigger-No third byte
3	Trigger-No forth byte (most significant byte)
4	Trigger-Type 1
5	Trigger-Type 2
6	CRC-8-CCITT (checksum)

Table 2.1: The transmission protocol to broadcast the trigger-ID to the FAD boards

A Cyclic Redundancy Check (CRC) over byte 0 - 5 is used to evaluate the integrity of the trigger-ID. An 8-CCITT CRC has been chosen which is based on the polynomial $x^8 + x^2 + x + 1$ (00000111, omitting the most significant bit). The resulting 1-byte checksum comprises the last byte of the trigger-ID. The transmission of the trigger-ID to the FAD boards is done by means of dedicated RS-485 buses (one per crate).

In the first byte of the trigger type indicator (see table 2.2) n0 - n5 indicate the number of trigger primitives required for a trigger, thus the 'n' of the 'n-out-of-40' majority coincidence. The two flags 'external trigger 1' and 'external trigger 2', when set, indicate a trigger from the corresponding NIM inputs. See also section 4.1 and table 4.11 for further information.

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
n5	n4	n3	n2	n1	n0	external trigger 2	external trigger 1

Table 2.2: Trigger-Type 1

The 'TIM source' bit in 'Trigger-Type 2' (see table 2.3) indicates the source of the timemarker signal: a '0' indicates the timemarker being produced in the FPGA while a '1' indicates the

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TIM source	LP_set_3	LP_set_2	LP_set_1	LP_set_0	pedestal	LP_2	LP_1

Table 2.3: Trigger-Type 2

timemarker coming from the clock conditioner. The flags 'LP_1' and 'LP_2' are set when the corresponding lightpulser has flashed while the 'pedestal' flag is set in case of a pedestal (random) trigger. An event having none of these flags set indicates a physics event. The bits 'LP_set_0' to 'LP_set_3' can be used in the future to code information about the light pulser settings. They only have a meaning in case of calibration events, but are not yet filled by the firmware (only zeros).

Chapter 3

FTM Commands

The communication between the FTM board and the FTMcontrol software, including the corresponding commands, protocols and data, is based on 16-bit words and big-endian. This is to facilitate the data-transmission over the Wiznet W5300 ethernet interface [4].

The basic structure of all commands is the same and given in table 3.1. After a start delimiter, the second word identifies the command. Next there is a parameter further refining the command, e.g. what to read. The fourth and fifth words are spares and should contain zeros only. Starting from the sixth word, an optional data block of variable size is following. This data block differs in length and content depending on command and parameter. In case of 'read' instructions, the corresponding data block is sent back.

So far eight different commands are foreseen: 'read', 'write', 'start run', 'stop run', 'ping FTUs', 'crate reset', 'autosend on/off' and 'config single FTU' (see table 3.2). The command parameters of the 'read' and 'write' commands are shown in table 3.3 and table 3.4, respectively. With the 'autosend on/off' command it is possible to switch off the automatic sending of trigger rates and error messages (see table 3.5).

The 'config single FTU' command has as its parameters a crate and a board number as it is shown in table 3.6. Before the 'config single FTU' command can be meaningfully executed, the corresponding data has to be written into the static data block (see section 4.1), using e.g. 'write single address' commands. The 'config single FTU' command then writes the new configuration to the corresponding FTU board, even if a run is ongoing.

In table 3.7 the parameters to start a run are listed. The type of the run is fully described in the FTM configuration (static data block, see section 4.1), which always has to be sent by the FTMcontrol before starting a run. Therefore the only option is to start an "endless" run or to take X events (Not yet implemented in firmware) instead. In the latter case X is defined by a two words (32 bit) long unsigned integer, making up the command data block. The 'start run' command enables the transmission of trigger signals (physics, calibration or pedestal) to the FAD boards and resets the trigger and time counters. There is no parameter for stopping a run. If a number of events has been specified ('take X events'), the run will terminate if either the 'stop run' command is received or the requested number of events is reached. In any case the trigger and time counters are reset, too.

In case of a 'ping FTUs' command the FTM will address the FTUs one by one and readout

word no	content
0	start delimiter ('@')
1	command ID
2	command parameter
3	spare: containing 0x0000
4	spare: containing 0x0000
5	data block (optional and of variable size)
...	...
X	data block

Table 3.1: FTM command structure

command-ID: bits									
15 ... 8	7	6	5	4	3	2	1	0	command
0	0	0	0	0	0	0	0	1	read
0	0	0	0	0	0	0	1	0	write
0	0	0	0	0	0	1	0	0	start run / take X events
0	0	0	0	0	1	0	0	0	stop run
0	0	0	0	1	0	0	0	0	ping all FTUs
0	0	0	1	0	0	0	0	0	crate reset
0	0	1	0	0	0	0	0	0	autosend on/off
0	1	0	0	0	0	0	0	0	config single FTU

Table 3.2: FTM command ID listing

command parameter: bits										
15 ... 8	7	6	5	4	3	2	1	0	command	data block
0	0	0	0	0	0	0	0	1	read complete static data block	no
0	0	0	0	0	0	0	1	0	read complete dynamic data block	no
0	0	0	0	0	0	1	0	0	read single address of static data block	address

Table 3.3: Command parameters for the 'read' command; only for the static data block single addresses can be read.

command parameter: bits										
15 ... 8	7	6	5	4	3	2	1	0	command	data block
0	0	0	0	0	0	0	0	1	write complete static data block	all configuration data
0	0	0	0	0	0	1	0	0	write single address of static data block	address + data

Table 3.4: Command parameters for the 'write' command; only the static data block can be written, therefore parameter value 0x2 is not used.

command parameter: bits										
15 ... 8	7	6	5	4	3	2	1	0	command	data block
0	0	0	0	0	0	0	0	0	reports disabled	no
0	0	0	0	0	0	0	0	1	reports enabled	no

Table 3.5: Command parameters for the 'autosend on/off' command

their DNA. The results are collected in the FTU list (see section 4.3), which is sent back to the FTMcontrol. There are no parameters for this command. With the 'crate reset' command the FTU and FAD boards of a particular crate are rebooted (only the FPGAs), where the command parameter defines the crate number (see table 3.8). Only one crate reset at a time is possible, i.e. the FTM firmware does not allow to reset multiple crates in one command.

command parameter: bits					
15 ... 11	10 ... 8	7 ... 2	1 ... 0	command	data block
0	slot no	0	crate no	configure FTU	no

Table 3.6: Command parameters for the 'config single FTU' command

command parameter: bits										
15 ... 8	7	6	5	4	3	2	1	0	command	data block
0	0	0	0	0	0	0	0	1	start run	no
0	0	0	0	0	0	0	1	0	take X events	number of events X

Table 3.7: Command parameters for the 'start run' command: "start run" means an "endless" run, i.e. no pre-defined number of events; if a number of events X is specified, this is done with a 32-bit unsigned long integer (big endian).

command parameter: bits										
15 ... 8	7	6	5	4	3	2	1	0	command	data block
0	0	0	0	0	0	0	0	1	reset crate 0	no
0	0	0	0	0	0	0	1	0	reset crate 1	no
0	0	0	0	0	0	1	0	0	reset crate 2	no
0	0	0	0	0	1	0	0	0	reset crate 3	no

Table 3.8: Command parameters for the 'crate reset' command: the command parameter may only contain a single "1" corresponding to only one crate reset at a time.

Chapter 4

FTM data blocks

The trigger master features two main data blocks, named 'static data block' and 'dynamic data block' in the following. They are implemented in the firmware as block-RAM. In addition, there is the so-called 'FTU list', which is filled only on request ('ping FTUs' command). If any of these blocks is sent to the FTMcontrol (either automatically or on demand), a header with a size of 14 words is added, and the whole data package is put between a start and an end delimiter (see table 4.1). The header is identical for all data blocks and contains solely read-only information: the type and length of the package, the FTM status, the FTM board ID (57-bit Xilinx device DNA [5, 6, 7, 8]), a firmware ID and the readings of the trigger counter and time stamp counter since last reset (see table 4.2).

start delimiter	header	data block	end delimiter
0xFB01	14 words	optional size	0x04FE

Table 4.1: Structure of a data package as sent by the FTM to the FTMcontrol software. The start and end delimiters are the same as used for the FAD boards.

In table 4.3 the encoding of the status of the FTM is shown.

4.1 Static data block

The static data block contains all the settings needed to configure and operate the FTM. It has to be written by the FTMcontrol each time before a run is started or, in general, some component has to be reprogrammed. In addition, whenever the FTM board receives a new static data block, it performs a complete reconfiguration including a reprogramming of the FTUs. As an exception, the clock conditioner is only reprogrammed, if its settings really changed. When a new static data block is received during a run, it is written to the memeory, but subsequently ignored. Table 4.4 summarizes the static data block. More details about the individual registers can be found in the subsequent tables.

word no	content	description
0x000	general settings	see table 4.5 and text
0x001	on-board status LEDs	see table 4.6

0x002	light pulser and pedestal trigger period	see table 4.7 and text
0x003	sequence of LP1, LP2 and PED triggers	see table 4.8 and text
0x004	light pulser 1 amplitude	see table 4.9 and text
0x005	light pulser 2 amplitude	see table 4.10 and text
0x006	light pulser 1 delay	8ns + delay value*4ns
0x007	light pulser 2 delay	8ns + delay value*4ns
0x008	majority coincidence n (for physics)	see table 4.11 and text
0x009	majority coincidence n (for calibration)	see table 4.11 and text
0x00A	trigger delay	8ns + delay value*4ns, 10 bits used
0x00B	timemarker delay	8ns + delay value*4ns, 10 bits used
0x00C	dead time	8ns + value*4ns, 16 bits used
0x00D	clock conditioner R0 bits 31...16	
0x00E	clock conditioner R0 bits 15...0	
0x00F	clock conditioner R1 bits 31...16	
0x010	clock conditioner R1 bits 15...0	
0x011	clock conditioner R8 bits 31...16	
0x012	clock conditioner R8 bits 15...0	
0x013	clock conditioner R9 bits 31...16	
0x014	clock conditioner R9 bits 15...0	
0x015	clock conditioner R11 bits 31...16	
0x016	clock conditioner R11 bits 15...0	
0x017	clock conditioner R13 bits 31...16	
0x018	clock conditioner R13 bits 15...0	
0x019	clock conditioner R14 bits 31...16	
0x01A	clock conditioner R14 bits 15...0	
0x01B	clock conditioner R15 bits 31...16	
0x01C	clock conditioner R15 bits 15...0	
0x01D	maj. coinc. window (for physics)	8ns + value*4ns, 4 bits used
0x01E	maj. coinc. window (for calibration)	8ns + value*4ns, 4 bits used
0x01F	spare	
0x020	enables patch 0 board 0 crate 0	see FTU documentation
0x021	enables patch 1 board 0 crate 0	see FTU documentation
0x022	enables patch 2 board 0 crate 0	see FTU documentation
0x023	enables patch 3 board 0 crate 0	see FTU documentation
0x024	DAC_A board 0 crate 0	see FTU documentation
0x025	DAC_B board 0 crate 0	see FTU documentation
0x026	DAC_C board 0 crate 0	see FTU documentation
0x027	DAC_D board 0 crate 0	see FTU documentation
0x028	DAC_H board 0 crate 0	see FTU documentation
0x029	Prescaling board 0 crate 0	(value+1)/2 [s], also autosend period
0x02A	enables patch 0 board 1 crate 0	see FTU documentation
0x02B	enables patch 1 board 1 crate 0	see FTU documentation
0x02C	enables patch 2 board 1 crate 0	see FTU documentation
0x02D	enables patch 3 board 1 crate 0	see FTU documentation
0x02E	DAC_A board 1 crate 0	see FTU documentation

0x02F	DAC_B board 1 crate 0	see FTU documentation
0x030	DAC_C board 1 crate 0	see FTU documentation
0x031	DAC_D board 1 crate 0	see FTU documentation
0x032	DAC_H board 1 crate 0	see FTU documentation
0x033	Prescaling board 1 crate 0	see FTU documentation
...	...	
0x1A6	enables patch 0 board 9 crate 3	see FTU documentation
0x1A7	enables patch 1 board 9 crate 3	see FTU documentation
0x1A8	enables patch 2 board 9 crate 3	see FTU documentation
0x1A9	enables patch 3 board 9 crate 3	see FTU documentation
0x1AA	DAC_A board 9 crate 3	see FTU documentation
0x1AB	DAC_B board 9 crate 3	see FTU documentation
0x1AC	DAC_C board 9 crate 3	see FTU documentation
0x1AD	DAC_D board 9 crate 3	see FTU documentation
0x1AE	DAC_H board 9 crate 3	see FTU documentation
0x1AF	Prescaling board 9 crate 3	see FTU documentation
0x1B0	active FTU list crate 0	see FTU documentation
0x1B1	active FTU list crate 1	see FTU documentation
0x1B2	active FTU list crate 2	see FTU documentation
0x1B3	active FTU list crate 3	see FTU documentation

Table 4.4: Overview of the FTM static data block

The FTM general settings register is detailed in table 4.5. The 'TIM_CLK' bit defines whether the time marker is generated by the FPGA ('TIM_CLK' = 0, default for physics data taking), or whether it is generated by the clock conditioner ('TIM_CLK' = 1, e.g. for DRS timing calibration). The 'ext_veto', 'ext_trig_1' and 'ext_trig_2' bits enable (1) or disable (0) the NIM inputs for the external veto and trigger signals, respectively. In order to select which trigger sources are active during a run, the bits 'LP1', 'LP2', 'ped' and 'trigger' are foreseen (0 disabled, 1 enabled). During a physics run, for example, 'LP1', 'ped' and 'trigger' should all be set to generate interleaved calibration and pedestal events as well as activate the 'n-out-of-40' trigger input. For a didicated pedestal run only 'ped' should be set, since in this case the FTM sends directly a trigger to the FADs. For calibration runs it depends on whether the external (LP1) or internal (LP2) light pulser is used: For the first case 'LP1' and 'trigger' have to be set, since here the full trigger chain is involved and the camera triggers based on G-APD signals. For the second case only 'LP2' is needed, because the shutter is closed and the FTM sends the trigger signal directly to the FADs (like for pedestal events). Bits 8 to 15 of the general settings register are not used up to now.

The 'on-board status LEDs' register shown in table 4.6 allows to switch a total of eight LEDs on the FTM board for debugging purposes by setting the corresponding bit high.

The period (time distance, see table 4.7), with which light pulser and pedestal triggers are sent, is stored in the register at address 0x002. It is given in [ms] and adjustable between 1 ms and 1023 ms (10 bits used). The next register defines the sequence of LP1, LP2 and pedestal events (see table 4.8).

In order to define the amplitude and characteristics of the light pulses that are generated by

word no	content	description
0x000	type of data package	1: SD, 2: DD, 3: FTU-list, 4: error, 5: single SD-word
0x001	length of data package	after header, including end delimiter
0x002	status of FTM	see table 4.3
0x003	board ID bits 63...48	FPGA device DNA
0x004	board ID bits 47...32	FPGA device DNA
0x005	board ID bits 31...16	FPGA device DNA
0x006	board ID bits 15... 0	FPGA device DNA
0x007	firmware ID	defined as a VHDL constant
0x008	trigger counter bits 31...16	at read-out time
0x009	trigger counter bits 15... 0	at read-out time
0x00A	time stamp bits 63...48	filled up with zeros
0x00B	time stamp bits 47...32	at read-out time
0x00C	time stamp bits 31...16	at read-out time
0x00D	time stamp bits 15... 0	at read-out time

Table 4.2: Header structure for sending a data block or error message, wherq 'SD' stands for the static data block and 'DD' for the dynamic data block.

status	value
IDLE_NOT_LOCKED	0x0001
CONFIG	0x0002
RUNNING_NOT_LOCKED	0x0003
IDLE	0x0101
RUNNING	0x0103

Table 4.3: Encoding of the status of the FTM board in the header. In this context 'LOCKED' means that the PLL of the Clock conditioner [2] is locked.

the LP1 and the LP2 system, the registers 'LP1 amplitude' and 'LP2 amplitude' are used, respectively. These registers are presented in table 4.9 and table 4.10. The two most significant bits allow to switch on additional LEDs, while the six least significant bits are used for the FM (frequency modulation) on the light pulser board. These six bits (FM1_5 ... FM1_0 and FM2_5 ... FM2_0, respectively) are frequency dividing factors and the resulting frequency for the feedback is $f_{FM} = \frac{5MHz}{25+|FM1\_5...FM1\_0|}$.

This FM signal is used for stabilizing the amplitude of the light pulses, see the schematics [11].

The light pulser systems are controlled from the FTM by means of four LVDS control lines: The first line goes directly to the LED driver circuit and triggers a lightpulse. The FM signal is on the second line. The third and forth line allow to switch on additional LEDs.

The different settings of the 'n-out-of-40' logic (physics or calibration events) are stored in two separate registers, which both have a structure according to table 4.11.

In addition, there are several registers in the static data block to define delays (e.g. for the trigger). Also a general dead time to be applied after each trigger can be set (to compensate for the delay of the busy line). The clock conditioner settings are specified at address 0x00D to 0x01C (LMK03000 from National Semiconductor, for more details see [2]).

Bit	15...8	7	6	5	4	3	2	1	0
Content	x	trigger	ped	LP2	LP1	ext_trig_2	ext_trig_1	ext_veto	TIM_CLK

Table 4.5: FTM general settings register

Bit	15...8	7	6	5	4	3	2	1	0
Content	x	red_3	red_2	gn_1	ye_1	red_1	gn_0	ye_0	red_0

Table 4.6: 'on-board status LEDs' register

Starting at address 0x020, the FTU settings are stored. The FTM always holds the complete FTU parameters in the static data block. For the meaning of these registers, please refer to the FTU firmware specifications document [10]. The register at address 0x029 is special in the sense that, in addition to its normal meaning, it also defines the time period with which the FTU rates are sent automatically to the FTMcontrol software. In case not all FTUs are connected during e.g. the testing phase, or a FTU is broken, the 'active FTU list' registers can be used to disable certain boards. Bits 9...0 of one of the active FTU lists (address 0x1B0 to 0x1B3, corresponding to crate 0 to 3) contain the "active" flag for every FTU board. Setting a bit activates the corresponding FTU board while a "0" deactivates it.

4.2 Dynamic data block

The dynamic data block shown in table 4.12 contains permanently updated data stored inside the FTM FPGA. It contains the actual on-time counter reading, the board temperatures (not yet supported) and the trigger rates measured by the FTUs. The on-board 12-bit temperature sensors are MAX6662 chips from Maxim Products. For more information about these components and their data see [9]. This data block is updated and sent periodically by the FTM. Thus the FTMcontrol software receives periodically a corresponding data package via ethernet. The counting interval of the FTU board 0 on crate 0 ('prescaling' register) defines the period. When sending the dynamic data block, the header defined in table 4.2 is added at the beginning.

Bit	15 - 10	9	8	...	2	1	0
Content	x	PERIOD_9	PERIOD_8	...	PERIOD_2	PERIOD_1	PERIOD_0

Table 4.7: Register for the period [ms] of calibration and pedestal events

Bit	15	14	...	10	9	...	5	4	...	0
Content	x	ped_S4	...	ped_S0	LP2_S4	...	LP2_S0	LP1_S4	...	LP1_S0

Table 4.8: Register defining the sequence of LP1, LP2 and pedestal events; 5 bits used per value. By setting e.g. LP1/LP2/PED = 3/2/1, the systems generates 3 LP1 triggers, followed by 2 LP2 triggers, followed by 1 PED trigger (if they are also activated in the 'general settings' register). The distance between the triggers is defined with another register (table 4.7).

Bit	15	14	13 ... 6	5 ... 0
Content	add_LEDs1_1	add_LEDs1_0	x	FM1_5 ... FM1_0

Table 4.9: Light pulser 1 amplitude register

Bit	15	14	13 ... 6	5 ... 0
Content	add_LEDs2_1	add_LEDs2_0	x	FM2_5 ... FM2_0

Table 4.10: Light pulser 2 amplitude register

Bit	15...6	5	4	3	2	1	0
Content	x	n5	n4	n3	n2	n1	n0

Table 4.11: Structure of the two majority coincidence (n-out-of-40) registers; the binary value in these registers is the minimum number n of FTU trigger primitives required to trigger an event (physics or calibration)

word no	content
0x000	on-time counter at read-out time bits 63...48, filled up with zeros
0x001	on-time counter at read-out time bits 47...32
0x002	on-time counter at read-out time bits 31...16
0x003	on-time counter at read-out time bits 15...0
0x004	temperature sensor 0: not used so far
0x005	temperature sensor 1: not used so far
0x006	temperature sensor 2: not used so far
0x007	temperature sensor 3: not used so far
0x008	rate counter bit 29...16 patch 0 board 0 crate 0
0x009	rate counter bit 15...0 patch 0 board 0 crate 0
0x00A	rate counter bit 29...16 patch 1 board 0 crate 0
0x00B	rate counter bit 15...0 patch 1 board 0 crate 0
0x00C	rate counter bit 29...16 patch 2 board 0 crate 0
0x00D	rate counter bit 15...0 patch 2 board 0 crate 0
0x00E	rate counter bit 29...16 patch 3 board 0 crate 0
0x00F	rate counter bit 15...0 patch 3 board 0 crate 0
0x010	rate counter bit 29...16 total board 0 crate 0
0x011	rate counter bit 15...0 total board 0 crate 0

0x012	Overflow register board 0 crate 0
0x013	CRC-error register board 0 crate 0
0x014	rate counter bit 29...16 patch 0 board 1 crate 0
0x015	rate counter bit 15...0 patch 0 board 1 crate 0
0x016	rate counter bit 29...16 patch 1 board 1 crate 0
0x017	rate counter bit 15...0 patch 1 board 1 crate 0
0x018	rate counter bit 29...16 patch 2 board 1 crate 0
0x019	rate counter bit 15...0 patch 2 board 1 crate 0
0x01A	rate counter bit 29...16 patch 3 board 1 crate 0
0x01B	rate counter bit 15...0 patch 3 board 1 crate 0
0x01C	rate counter bit 29...16 total board 1 crate 0
0x01D	rate counter bit 15...0 total board 1 crate 0
0x01E	Overflow register board 1 crate 0
0x01F	CRC-error register board 1 crate 0
...	...
0x1E7	CRC-error register board 9 crate 3

Table 4.12: FTM dynamic data block

4.3 FTU list

When the FTM board receives the 'ping all FTUs' instruction, it sends a ping command to all FTU boards and gathers the FTU boards responses to a list. This list is called 'FTU list' and shown in table 4.13. When the FTU list is complete, it is sent back via ethernet with the header defined in table 4.2.

address	content
0x000	total number of responding FTU boards
0x001	number of responding FTU boards belonging to crate 0
0x002	number of responding FTU boards belonging to crate 1
0x003	number of responding FTU boards belonging to crate 2
0x004	number of responding FTU boards belonging to crate 3
0x005	active FTU list crate 0
0x006	active FTU list crate 1
0x007	active FTU list crate 2
0x008	active FTU list crate 3
0x009	address of first FTU board and number of sent pings until response
0x00A	DNA of first FTU board bit 63 ... 48
0x00B	DNA of first FTU board bit 47 ... 32
0x00C	DNA of first FTU board bit 31 ... 16
0x00D	DNA of first FTU board bit 15 ... 0
0x00E	CRC error counter reading of first FTU board
0x00F	address of second FTU board and number of sent pings until response
0x010	DNA of second FTU board bit 63 ... 48
0x011	DNA of second FTU board bit 47 ... 32

0x012	DNA of second FTU board bit 31 ... 16
0x013	DNA of second FTU board bit 15 ... 0
0x014	CRC error counter reading of second FTU board
...	...
0x0F8	CRC error counter reading of last FTU board

Table 4.13: FTU list

In case there is no response to a 'ping' for a certain FTU address, there are up to two repetitions. If there is still no answer, only zeros are written into the FTU list for this particular board. A responding FTU board gets a regular entry, including the number of 'ping' sent until response. The number of pings is coded together with the FTU board address as shown in table 4.14. The two bits 'pings_0' and 'pings_1' contain the number of 'pings' until response of an FTU board (coded in binary). The 'DNA' of the FTU board is the device DNA [5, 6, 7, 8] of the FPGA on the responding FTU board. This is a unique 57 bit serial number unambiguously identifying every Xilinx FPGA. In the most significant word (bit 63 ... 48) bits 63 down to 57 are filled with zeros.

Bit	15 ... 10	9	8	7	6	5	...	0
Content	x ... x	pings_1	pings_0	x	x	A5	...	A0

Table 4.14: Address of FTU board and number of pings until response. In case there is no response at all, this number is set to 0.

Chapter 5

FTU communication error handling

When the FTM board is communicating with a FTU board via RS-485, the FTU board has to respond within 2 ms (after the last byte was transmitted). If this timeout expires, or the response sent back by the FTU board is incorrect, the FTM resends the datapacket after the timeout. If this second attempt is still unsuccessful, a third and last attempt will be made by the FTM board. An error message will be sent to the FTMcontrol whenever a FTU board does not send a correct answer after the first call by the FTM board. This message (see table 5.1) contains, after the standard header (see table 4.2), the number of calls until response (0 if no response at all), and the corresponding data packet which was sent to the FTU board. In order to avoid massive error messages for e.g. test setups with single FTUs, the 'active FTU list' can be employed to disable FTUs from the bus. In that case the FTM will not try to contact the corresponding boards.

word no	content
0x000	number of calls until response (0 if no response at all)
0x001 ... 0x01C	slow control data packet sent to FTU (28 words/bytes)

Table 5.1: FTU communication error message (after standard header); for a description of the FTU data package, see [10].

Bibliography

[1] Paul Scherrer Institut PSI. *DRS4 9 Channel, 5 GSPS Switched Capacitor Array.* http://drs.web.psi.ch datasheet.

[2] National Semiconductor Corporation. *LMK03000 Family Precision Clock Conditioner with integrated VCO*, 2008. datasheet.

[3] ETH Zürich, IPP. *FTM Schematics*, 2010.

[4] WIZnet Co.Ltd. *W5300 Fully Hardwired Network protocol Embedded Ethernet Controller*, 2008. datasheet.

[5] Xilinx. *Spartan-3AN FPGA Family Data Sheet*, 2009.

[6] Xilinx. *Spartan-3A DSP FPGA Family: Data Sheet*, 2009.

[7] Xilinx. *Advanced Security Schemes for Spartan-3A/3AN/3A DSP FPGAs*, 2007.

[8] Xilinx. *Security Solutions Using Spartan-3 Generation FPGAs*, 2008.

[9] Maxim Integrated Products. *12-Bit plus Sign Temperature Sensor with SPI-Compatible Serial Interface MAX6662*, 2001. datasheet.

[10] ETH Zürich, IPP. *FTU Firmware Specifications v3*, 2010.

[11] ETH Zürich, IPP. *FLD Schematics, FACT light driver*, 2010.