A TDC for the characterization of KM3NeT PMTs

A. Zwart[a], E. Heine[a], J. Hogenbirk[a], P. Jansweijer[a], G. Kieft[a], S. Mos[a], E. de Wolf[a,b,*],
on behalf of the KM3NeT Consortium

[a] *Nikhef, Science Park 105,1098 XG Amsterdam, The Netherlands*
[b] *University of Amsterdam, Science Park 904,1098 XH Amsterdam, The Netherlands*

Abstract

The optical modules of the future KM3NeT neutrino telescope will contain many photomultiplier tubes with a diameter of about three inch. In order to characterize these photomultiplier tubes, a 16 channel Time-Over-Threshold TDC with a GigaBit Ethernet communication channel has been built in an Altera StratixIV evaluation board. The TDC data are packed in UDP packages and sent to the host PC. Control is implemented using I^2C command packages send to the TDC by the host PC. After execution of I^2C commands a result package is send back to the host. We will present the TDC setup and first results.

Keywords: KM3NeT, PMT, TDC
PACS: 95.55.-n, 95.85.Ry, 85.60.Ha

1. Introduction

The KM3NeT digital optical module (DOM) will contain many small (3 inch) photomultipliers (PMTs). To characterize these PMTs a test setup of a 16 channel PMT readout has been built, using a Stratix IV GX FPGA Development Kit with two mezzanine boards, a Terasic SFP HSMC Board and a Terasic HSTC to GPIO Daughter Board. Signals from every PMT are amplified and discriminated [1] to generate Time Over Threshold (TOT) signals. These signals are sampled with a high frequency clock in order to achieve a digital representation of the Time Over Threshold. The timestamps of active TOT signals are packed in a IPv4 UDP packet and sent via Ethernet to a specified Port. The readout can be controlled via a separate Ethernet Port. We will describe the firmware for the FPGA.

2. Architecture

The architecture of the test setup is shown in Fig. 1. The TDC consists of one *Coarse Time* (Fig. 2) counter that is clocked by a global clock for all channels and a four bit Serdes for the *Fine Time* per channel. To remove noisy channels the inputs of the TDC can be disabled by a mask register in the I^2C slave. The *Zero-Suppress* module checks the *Fine Time* data for each *Coarse Time* period and only the *Fine Time* of the leading and the trailing edge of the input pulse are sent to the *Packet Builder*. The *Gigabit Ethernet* module consists of licensed IP Cores from Altera.

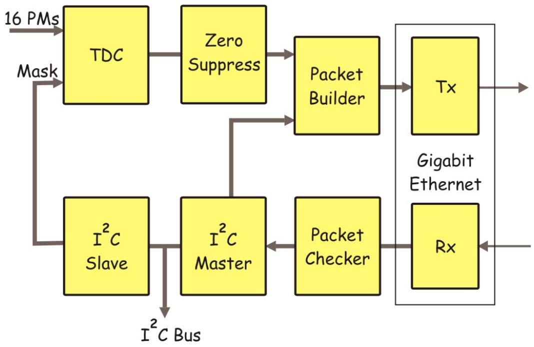


Figure 1: Architecture of the test setup.

The *I^2C Master* module is controlled by Ethernet. It can read and write registers both inside and outside the FPGA.

2.1. TDC

The 16 channel TDC (Fig. 2) is not a traditional TDC with a start and stop, but samples the input PMT signals to measure the Time Over Threshold [2]. It consists of a 24-bit *Coarse Time* counter and 4-bit shift registers (serdes) for the *Fine Time*. The clock of the shift registers and that of the *Coarse Time* are both supplied by a *Phase Loop Lock (PLL)*, to keep the the phase relation between them fixed. The frequency of the clock of the shift registers is four times higher than that of the *Coarse Time (CT)* clock, so the *Coarse Time* period is divided into four phases to allow for a *Fine Time* step of $\frac{1}{4}t_{CT}$. Fig. 3 shows the functional diagram. For synchronization purposes, a local offset of the *Coarse Time* counter is implemented, which is loaded into the counter on the arrival of a global reset or a *Coarse Time* counter reset. Its value can be set

*Corresponding author
Email address: `e.dewolf@nikhef.nl` (E. de Wolf)

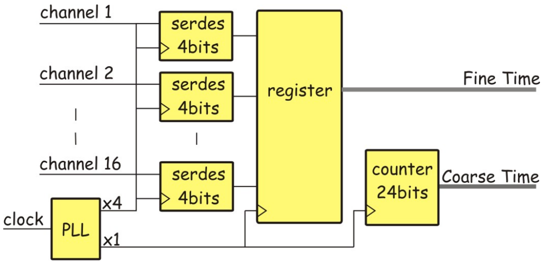


Figure 2: The 16 Channel TDC.

via the I^{2}C interface.

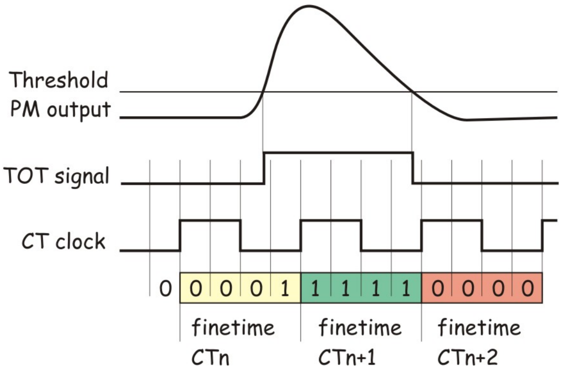


Figure 3: Functional diagram.

2.2. Zero Suppression

Zero Suppression is achieved by an n-stage pipelined processor, where n. is the number of channels. Fig. 4 shows a graphical representation of the zero-suppress mechanism for the example of four input channels. The processor is clocked by the *Coarse Time* clock. Every stage has two registers: one with the processed data and an other with data to be processed. Processing of the data consists of checking the *Fine Time*: if it is 0000 and the *Fine Time* of the former *Coarse Time* period is not equal to 1111 the data is dismissed; if it is 1111 and the F*Fine Time* of the former *Coarse Time* period is not equal to 0000 the data is also dismissed; in all other cases data is and the *Channel number* is appended to the "processed data register" of the next stage and the rest of the channel *Fine Time* data is copied in the "to be processed register" of the next stage. The result is that if the *Fine Time* of a channel is zero, no data of that channel are written in the event data.

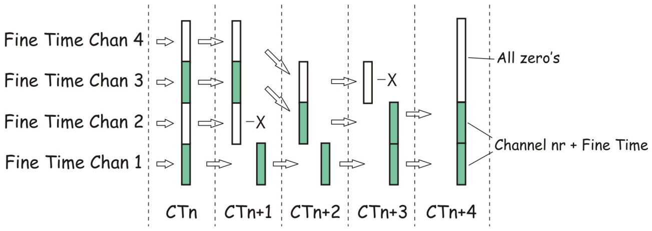
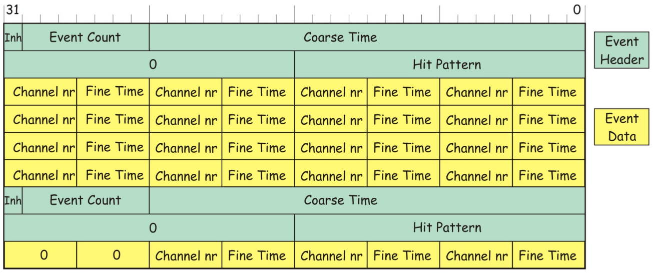


Figure 4: Zero suppression in the example of four input channels.

2.3. Event Format



31							0
Inh	Event Count	Coarse Time					
0				Hit Pattern			
Channel nr	Fine Time	Channel nr	Fine Time	Channel nr	Fine Time	Channel nr	Fine Time
Channel nr	Fine Time	Channel nr	Fine Time	Channel nr	Fine Time	Channel nr	Fine Time
Channel nr	Fine Time	Channel nr	Fine Time	Channel nr	Fine Time	Channel nr	Fine Time
Channel nr	Fine Time	Channel nr	Fine Time	Channel nr	Fine Time	Channel nr	Fine Time
Inh	Event Count	Coarse Time					
0				Hit Pattern			
0	0	Channel nr	Fine Time	Channel nr	Fine Time	Channel nr	Fine Time

Figure 5: Event format.

Data are transmitted in a format containing two 32-bit headers followed by the fine time and channel data of m times 32-bit words, where m depends on the number of channels with valid data. The header consists of a 16-bit *Hit Pattern*, a 24-bit *Coarse Time*, a 7-bit incremental *Event Count* and an *Inhibit* bit. This is illustrated in the table in Fig. 5. The back-end will use the contents of the *Hit Pattern* field to extract the data and can also monitor the *Inhibit* bit or the increment of the *Event Counter* field for missing events and the *Coarse Time* counter field of calibration events as a cross-check of the synchronization.

2.4. I^2C

Master The I^2C *Master* is compatible with the I^{2}C-bus Specification [3] and can transfer data at rates between 100 kbit/s and a theoretical maximum of 25 Mbit/s depending on the I^2C *Clock Divider Register* in the I^2C *Slave* interface. The I^2C *Master* can access registers of variable length or execute single byte transfers (Fig. 6).

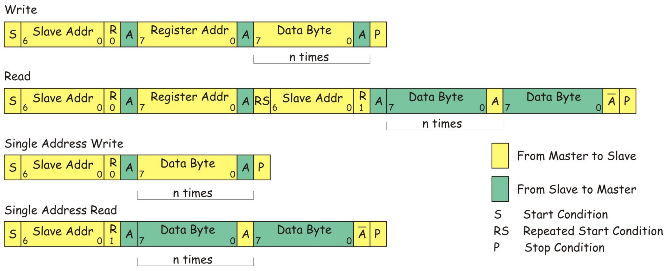
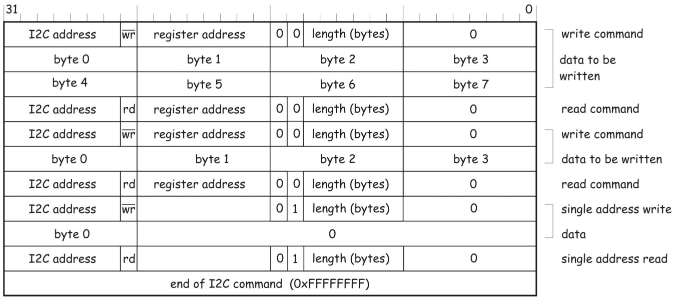

Write
S Slave Addr R A Register Addr A Data Byte A P
n times
Read
S Slave Addr R A Register Addr A RS Slave Addr R A Data Byte A Data Byte A P
n times
Single Address Write
S Slave Addr R A Data Byte A P
n times
Single Address Read
S Slave Addr R A Data Byte A Data Byte A P
n times
From Master to Slave
From Slave to Master
S Start Condition
RS Repeated Start Condition
P Stop Condition

Figure 6: I^{2}C cycles.

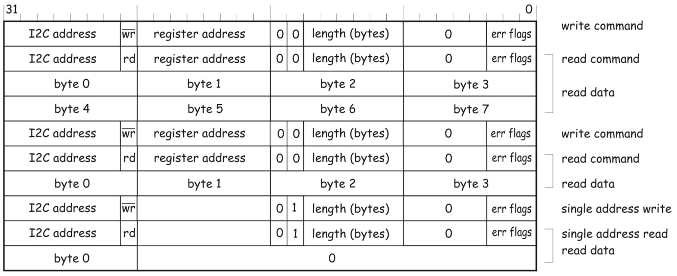
The I^2C *Master* receives a series of commands terminated by an end of command string from the *Gigabit Ethernet* interface. After execution of the commands, the data and commands with error flags are sent back to the Ethernet interface. The table in Fig. 7 shows the I^{2}C command format. The commands sent back to the Ethernet interface are shown in the table in Fig. 8. In case of a write command, only the command with the error flags is sent. In the case of a read command, the read command with error flags is sent first followed by the data. Error flag(0) means: no acknowledge after the address cycle; error flag(1) means: no acknowledge after a register address



31						0	
I2C address	wr	register address	0	0	length (bytes)	0	write command
byte 0		byte 1			byte 2	byte 3	data to be written
byte 4		byte 5			byte 6	byte 7	
I2C address	rd	register address	0	0	length (bytes)	0	read command
I2C address	wr	register address	0	0	length (bytes)	0	write command
byte 0		byte 1			byte 2	byte 3	data to be written
I2C address	rd	register address	0	0	length (bytes)	0	read command
I2C address	wr		0	1	length (bytes)	0	single address write
byte 0		0					data
I2C address	rd		0	1	length (bytes)	0	single address read
end of I2C command (0xFFFFFFFF)							

Figure 7: I^{2}C command format.

cycle; and error flag (2) means: no acknowledge after a data write cycle.



31							0	
I2C address	wr	register address	0	0	length (bytes)	0	err flags	write command
I2C address	rd	register address	0	0	length (bytes)	0	err flags	read command
byte 0		byte 1			byte 2	byte 3		read data
byte 4		byte 5			byte 6	byte 7		
I2C address	wr	register address	0	0	length (bytes)	0	err flags	write command
I2C address	rd	register address	0	0	length (bytes)	0	err flags	read command
byte 0		byte 1			byte 2	byte 3		read data
I2C address	wr		0	1	length (bytes)	0	err flags	single address write
I2C address	rd		0	1	length (bytes)	0	err flags	single address read read data
byte 0		0						

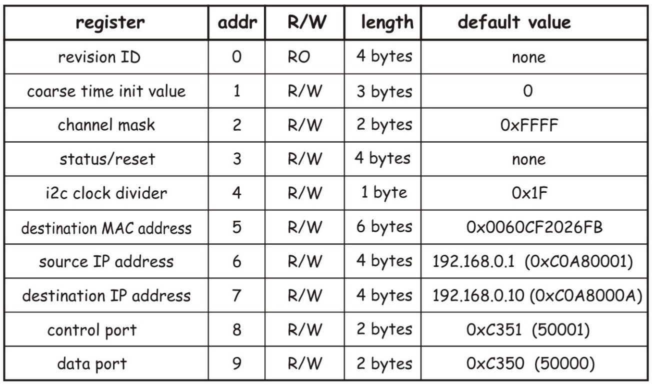
Figure 8: I^{2}C results format.

2.5. I^{2}C Slave

The *I^{2}C Slave* interface is compatible with the I^{2}C - bus Specification [3] and can transfer data at rates up to 1 Mbit/s. The *I^{2}C Slave* can access the 10 registers that control the PMT readout. The I^{2}C address of the Slave is programmed to $0x20$, the register addresses are defined in the table in Fig. 9. Bytes 3,2 and 1 of *Revision ID* represent the date in $YYMMDD$ format. Byte 0 is the revision number. The status/reset register has two modes. When data is written to the register, resets are generated: bit 0 generates a "global reset", bit 1 generates a "*Coarse Time* counter reset" and bit 2 causes a reset of the status registers. When the status/reset register is read out, bits 23 down to 0 represent the number of packets sent and bits 31 down to 24 show the number of *Coarse Time* periods with an active inhibit ($0xFF$ means overflow).

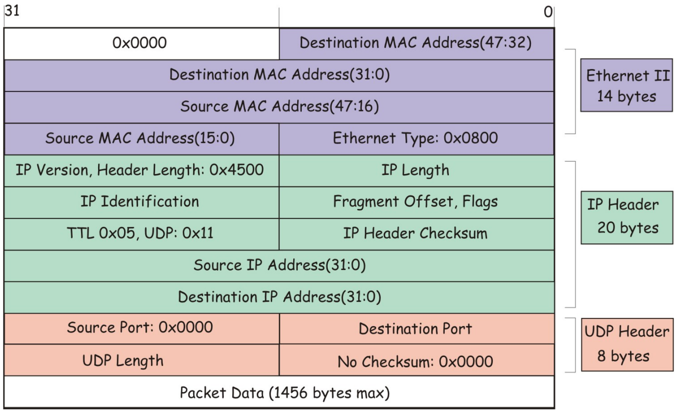
2.6. Packet Builder

The Packet Builder receives the data packets from both the *Zero Suppress* module and the *I^{2}C Master*. Of these two, the *I^{2}C Master* has a higher priority because its packets are shorter and occur less frequent. After receiving a packet from the *I^{2}C Master*, the *Packet Builder* sends the *Ethernet Header* to the MAC first followed by the data packet. The destination port in the header is the *Destination Control Port* (see the table in Fig. 10). When the *Packet Builder* receives event data from the *Zero Suppress* module it collects events until the data volume reaches the maximum of 1448 bytes. Following this, the Ethernet Header is sent to the MAC followed by the data of the events (max. 120 events). The destination port in the header is the *Destination Data Port* (see the table in Fig. 10).



register	addr	R/W	length	default value
revision ID	0	RO	4 bytes	none
coarse time init value	1	R/W	3 bytes	0
channel mask	2	R/W	2 bytes	0xFFFF
status/reset	3	R/W	4 bytes	none
i2c clock divider	4	R/W	1 byte	0x1F
destination MAC address	5	R/W	6 bytes	0x0060CF2026FB
source IP address	6	R/W	4 bytes	192.168.0.1 (0xC0A80001)
destination IP address	7	R/W	4 bytes	192.168.0.10 (0xC0A8000A)
control port	8	R/W	2 bytes	0xC351 (50001)
data port	9	R/W	2 bytes	0xC350 (50000)

Figure 9: Register addresses.



31	0	
0x0000	Destination MAC Address(47:32)	Ethernet II 14 bytes
Destination MAC Address(31:0)		
Source MAC Address(47:16)		
Source MAC Address(15:0)	Ethernet Type: 0x0800	
IP Version, Header Length: 0x4500	IP Length	IP Header 20 bytes
IP Identification	Fragment Offset, Flags	
TTL 0x05, UDP: 0x11	IP Header Checksum	
Source IP Address(31:0)		
Destination IP Address(31:0)		
Source Port: 0x0000	Destination Port	UDP Header 8 bytes
UDP Length	No Checksum: 0x0000	
Packet Data (1456 bytes max)		

Figure 10: Ethernet header.

2.7. Packet Checker

The Packet Checker checks in the Ethernet Header the destination MAC address and the destination port. If both are OK then the data is sent to the I^{2}C Master.

2.8. Gigabit Interface

The Ethernet interface consists of two Altera IP cores [4] - the "1000Mb Small Mac" and the "1000BASE- X/SGMII PCS only" - and a state machine to initialize the cores. The "1000Mb Small Mac" has a 32-bit wide FIFO interface. The "1000BASE-X/SGMII PCS only" core is interfaced to the MAC via the 8-bit GMII bus. The SGMII output of the PCS is directly fed to the SFP channel 4 of the Terasic SFP HSMC Board.

2.9. First results

To validate the correct working of the TDC a 25 ns pulse from a pulse generator is used as a simulated Time

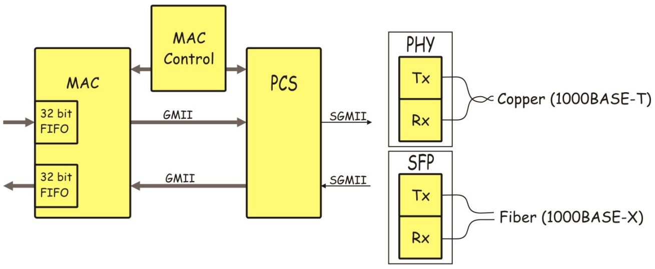
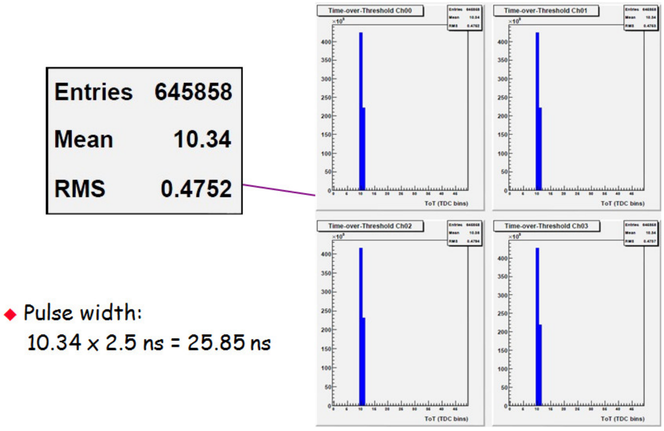


Figure 11: Ethernet interface.

over Threshold PMT signal. The frequency of the course clock of the TDC is set to 100 MHz resulting in a bin size of 2.5 ns. The histograms in Fig. 12 show that the pulse width measured by the TDC (25.85 ns) corresponds well with that from the pulse generator. In Fig. 13 an overview is shown of the test set-up used to readout 16 PMTs in a hemisphere of a KM3NeT DOM. The prototype DOM with the 16 PMTs is placed in a dark box with a led pulser that generates light pulses controlled by a pulse generator. The intensity of the led pulser is controlled via the settings of the pulse width and pulse amplitude of the pulse generator. Fig. 14 shows the measured Time over Threshold for one PMT, measured using a led pulser driven by a 30 ns pulse (right) and driven by a 40 ns pulse (left).



Entries	645858
Mean	10.34
RMS	0.4752

- Pulse width:
  10.34 x 2.5 ns = 25.85 ns

Figure 12: 25 ns pulse from pulse generator.

2.10. *Summary and Outlook*

For the characterization of photomultipliers in a KM3NeT DOM, a sixteen channel Time over Threshold TDC with Ethernet Readout/Control is realized in an Altera FPGA evaluation board equipped with a Stratix IV GX FPGA. Extension of the TDC to a 31 channel TDC for readout of a full KM3NeT DOM is straightforward. Several other developments and improvements are foreseen, among them a higher resolution by increasing the frequency of the *Course Clock*.

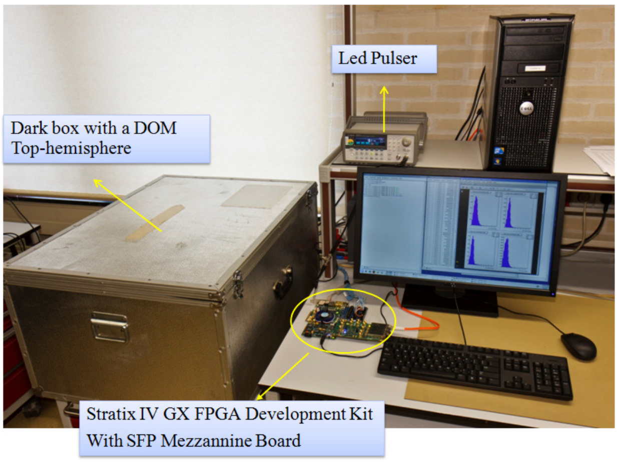
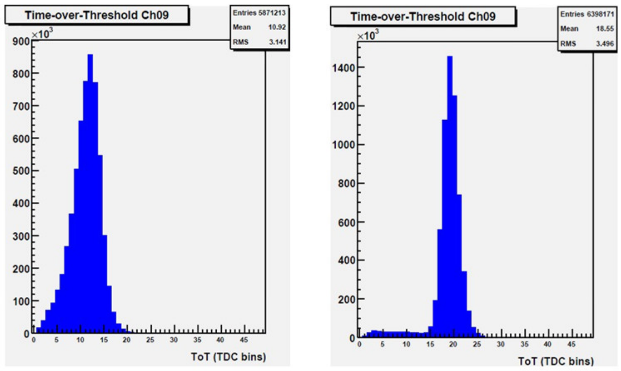


Figure 13: Test setup.



- Plots of a PMT with led-flasher
  - Right plot: led pulse of 30 ns (mean = 10.92 bin)
  - Left plot: led pulse of 40 ns (mean = 18.55 bin)

Figure 14: Test results PMTs.

3. Acknowledgment

This work is part of the research programme of the 'Stichting voor Fundamenteel Onderzoek der Materie (FOM)', which is financially supported by the 'Nederlandse Organisatie voor Wetenschappelijk Onderzoek (NWO)'. This study was supported by the European Comission through the FP6 KM3NeT Design Study Contract No. 011937 and the FP7 KM3NeT Preparatory Phase Grant Agreement No. 212525.

References

[1] A Front end ASPIC for the readout of the PMT in the KM3NeT detector. Derk Janenna et al., Proceedings TWEPP 2010 http://iopscience.iop.org/1748-0221/5/12/C12040 (2012)

[2] A 96-channel FPGA-based Time-to-Digital Converter (TDC) and fast trigger processor module with multi-hit capability and pipeline. Mircea Bogdan et al. Nuclear Instruments and Methods in Physics Research A554 (2005) 444-457

[3] The I^{2}C-Bus Specification Version 2.1 January 2000

[4] Altera Triple Speed Ethernet Megacore Functional User Guide Version 9.1 November 2009