TOR user manual

Lijiao Liu and Dong Wang

Created: October 2011
Last Update: 22th October 2011
Draft 0.1

Contact: Lijiao Liu
Institute of Physics and Technology
University of Bergen
Allégaten 55
5007 Bergen
Norway

Email: Lijiao.Liu@ift.uib.no
Private email: liulijiaollj@gmail.com

Contents

Chapter 1

Firmware description

1.1 Functionality

The task of the *Trigger-OR* (TOR) is to logically OR all local *Level-0* (L0) triggers coming from the *Trigger Region Unit* (TRU)s, combining them into only one L0 trigger on the *Central Trigger Processor* (CTP) side for the event, and generate L1 triggers on the basis of digitized Analog-sum signals coming from TRUs. L1 trigger includes one basic trigger and two advanced triggers: a total transverse energy trigger and an isolated photon trigger. Therein, the basic trigger and the total energy trigger are generated based only on the energy information; whereas the isolated photon trigger depends on both energies of photons and associated addresses. The basic trigger consists of three outputs: *Level-1 Low* (L1L), *Level-1 Middle* (L1M), and *Level-1 High* (L1H). The other two triggers have only one output for each. Nevertheless, there are only three *Level-1* (L1) outputs physically connected to the CTP, which results in a selection of triggers as outputs to the CTP on demand.

L1 triggers are generated by the TOR.

Figure 1.1 gives the block diagram of the firmware of the TOR. The *Bus controller* module is the *Detector Control System* (DCS) interface in the TOR firmware to act as a DCS slave. The DCS interface always acknowledges a transaction as long as the command is mapped to the address space of the TOR firmware. The *Register controller* module contains registers for the DCS board to control the configuration of the trigger output logic, the generation of L0 and L1 triggers, and to monitor some parameters of triggers as well. The trigger information is received by the *TTC Receiver* (TTCrx) chip on the DCS board via two lines, which are directly routed to the *Field Programmable Gate Array* (FPGA) on the TOR without any post-processing in the TTCrx. Therefore, a trigger decoder module is needed to decode the trigger information. Because only Confirmed-L0 is required for L1 generation, the *Trigger decode module* handles only Channel A that consists of Confirmed-L0 and L1 information. The *Data receiver* module is in charge of receiving the data related to Analog-sums from TRUs. The

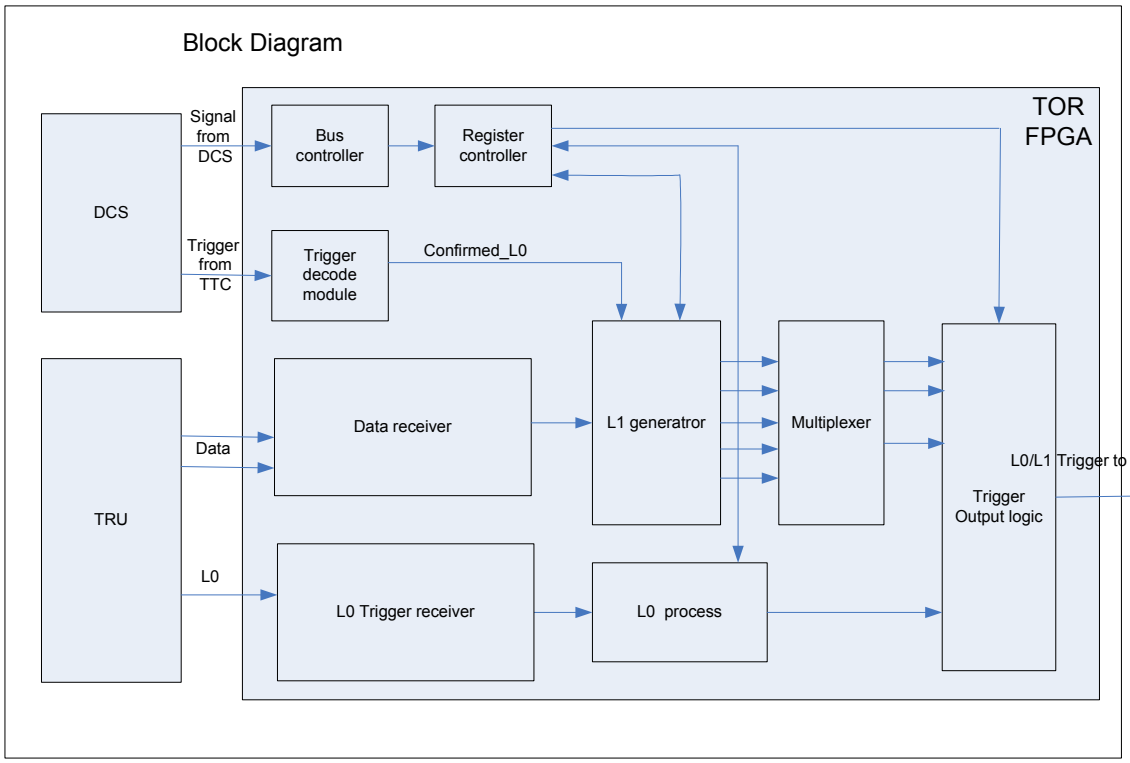

Figure 1.1: Block diagram of the TOR firmware.

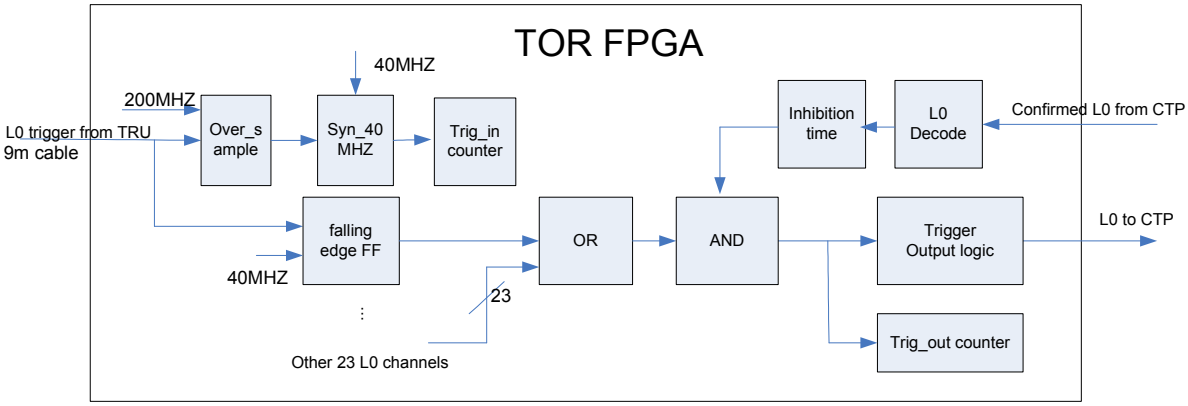

Figure 1.2: The process of L0 triggers in the TOR.

process of L0 in the TOR is shown in Figure 1.2. Each trigger input channel has its own processing channel (*Oversample, Syn_40M, Trig_in_counter, falling_edge_FF*). The *Trigger output logic* is designed for each trigger output according to the requirement of the CTP.

The manual describes the firmware used at P2 currently, which focuses on the L0 generation.

1.2 L0 calculation in the TOR

The *Oversample* module makes sure the L0 is clocked in correctly, even if there is a phase shift for the clock in the TOR relative to that in the TRUs. The output of the *Oversample* module is clocked at 200 MHz, and it is necessary to resynchronize the 5 ns (200 MHz) trigger to the 40 MHz domain. Plus resynchronize, two clock cycles (40 MHz) are needed. So in order to reduce timing consumption, the *Oversample* module is only used for the *Trig_in counter*.

The clock is distributed from the *Local Trigger Unit* (LTU) to the trigger electronics. The optical fibers from the LTU to all DCS boards (only two are shown in the figure) are the same for the *PHOton Spectrometer* (PHOS) detector, therefore, the clock outputs from DCS boards are aligned. The TRU gets its clock via the *Gunning Transceiver Logic* (GTL) bus. Relative to the clock of TOR, it has a delay of approximate 3 ns as tested. All cables between the TRUs and the TOR have a length of 9 meters, resulting in a delay of around 48 ns. So the signals from TRUs, driven by the rising edge of the clock, arrive 51 ns (around 2 clock cycles) relative to the rising edge of the TOR clock. Therefore, the trigger inputs are registered in the TOR at the falling edge of the clock to avoid extending the length of triggers.

During the first trigger commissioning at *Point 2* (P2), there were thousands of unexpected L0 triggers during reading out *Front End Card* (FEC)s since the access of ALTROs induces noise. This is now avoided by defining some inhibit time, called *dead time* of a trigger, for L0 after the TOR receives the Confirmed-L0 from the trigger system. The TOR does not send any triggers to the CTP when reading out FECs.

The L0 triggers coming from TRUs are counted in the *Trig_in counter* modules for each trigger input channel. The values in the *Trig_in counter* are refreshed every 1 seconds[1]. Also there is a *Trig_out counter* for the L0 before it is sent to the CTP, it is refreshed every 1 second[2]. These two types of counters are designed for monitoring and commissioning. There should be also counters to record the number of triggers per run, it is not implemented in this firmware.

The L0 trigger is finally output to the CTP by the *Trigger Output logic* module, registered by the rising edge of the clock. So in total, only one clock cycle is consumed for the trigger

[1] It records the number of triggers in 1 seconds.
[2] It records the number of triggers in 1 seconds.

process in the TOR. The *Trigger Output logic* module is specified by the CTP group. It is suitable for all trigger outputs.

1.3 Trigger output logic

All triggers are transmitted to the CTP in an uniform format, as defined in a detailed specification by the CTP group. There are many triggers going into the CTP, so each trigger should have its own signature. The signature is used to identify trigger signals, to ensure that the trigger signals are connected to the right inputs of the CTP, to assess the quality of cable connection and to measure the bit error rate. When there is no real trigger fed into the CTP, it needs a random trigger for debugging and testing. In addition, a toggling trigger mode of interval '0' and '1' is performed to synchronize the inputs with the *Large Hadron Collider* (LHC) clock [1]. As a result, there are 4 modes in the front end trigger logic, which is performed on the TOR board for PHOS. The selection of trigger input modes (normal, toggle, signature, etc.) is controlled by the CTP software via the *Distributed Information Management* (DIM) server [2].

The trigger signature data stream consists of a common 8-bit header (B"10110001") and the input-specific 14-bit signature code, where the first 7 bits of the code are the unique identifier of the trigger inputs, and the second 7 bits are the 1's complement of all first 7 bits. The trigger inputs from PHOS are allocated with signature number ranging from 4 to 9, therein 4, 5, 6, 7 are used for L0 trigger, L1L, L1M, and L1H respectively. The signature data stream is repeated in intervals of about 25 μs according to the requirement of the CTP group.

As shown in Figure 1.3, the signature data stream is implemented with a 22-bit parallel-in-serial-out shift register. A 10-bit counter is started when the parallel signature is loaded, ensuring that the data stream is sent out at intervals of 25 μs. The random trigger is supposed to have a programmable rate, which can be implemented based on a 31-bit *Linear Feedback Shift Register* (LFSR) and a comparator. The LFSR generates a random pattern synchronous with the clock. The average rate of the random trigger output is dependent on the programmable content of the 31-bit register. Only when the content of the 31-bit LFSR is smaller than the content of the 31-bit register, a trigger pulse is generated. Trigger pulses are distributed pseudo-randomly, actually they are repeated approximately every 53 s corresponding to $2^{31} - 1$ clock intervals. The toggling trigger is a pattern of alternating ones and zeros, which can be easily implemented by reversing the input clock by an inverted flip-flop. The default option is Normal operation, the other three modes are enabled only when the specific option code is set accordingly in order to save power consumption.

The option mode can be selected either by the CTP via TIN-proxy [3], which is a software based on DIM, or by PHOS operators manually via the DCS board on the TOR. In both ways,

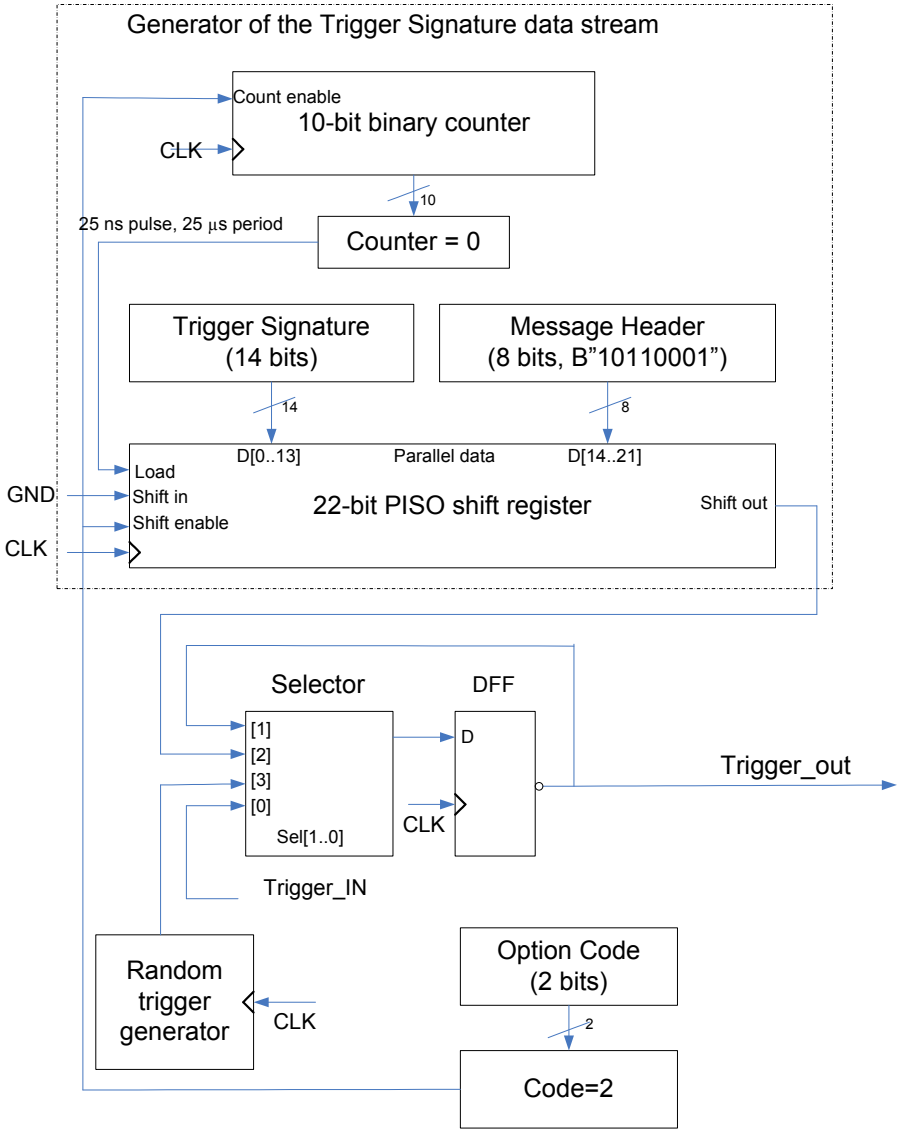

Figure 1.3: The block diagram of trigger output logic [2] (edited).

the selection is done by writing the option code register, after the corresponding configuration of each mode is set to the right state via the the DCS board.

The trigger output logic block is related to one single trigger output. As a result, there are four trigger output logic blocks implemented in the TOR owing to four trigger outputs of the PHOS.

1.4 Basic L1 generation

Only the local maximum 4×4-sum is needed for the basic L1 trigger. In the current firmware, the local three levels of basic L1 triggers are generated in the TRUs (Figure 1.4). In the

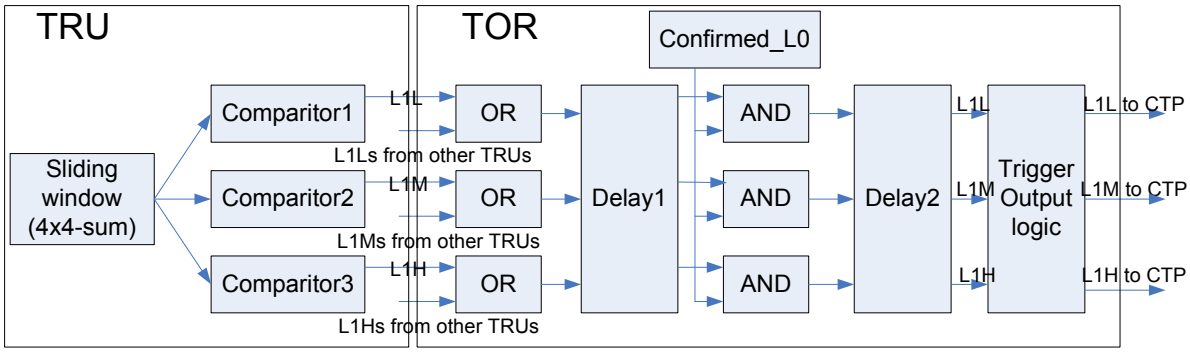

Figure 1.4: The block diagram of the basic L1 trigger.

TRUs, the same sliding window channels (4×4-sums), used for L0 generation, is compared with three thresholds to get basic L1 triggers. In the TOR, the L1 triggers from all TRUs are logically ORed first of all. After the OR operation, L1 triggers are ready within 800 ns after the interaction, whereas the Confirmed-L0 arrives at the TOR within 1.2 μs. Therefore a L1 trigger is delayed by two SRL16s (shift register *Look Up Table* (LUT)s) before it is logically ANDed with the Confirmed-L0 as shown in the figure. L1 triggers are supposed to arrive the CTP 6.1 μs after the interactions, but the L1 triggers after AND operation are available within 1.3 μs after the interactions. A long delay is required and a *Random Access Memory* (RAM) block is used to implement it. The RAM is a simple dual port RAM, 4-bit wide for both read and write operations. Three bits out of four are allocated to L1 triggers (L1L, L1M and L1H).

Chapter 2

Registers description

2.1 Registers related to the trigger output logic

The following registers set the test mode for each trigger output to the CTP. As described previously, four outputs are available, one for L0, three for basic L1 triggers: L1L, L1M, L1H. Tables 2.1 2.2 2.3 **??** give the register description. Trig*_OptionCode selects a mode among toggling, random, normal and signature. Trig*_Signature gives the signature. Trig*_MessageHeader is the common head of signatures. Trig*_Prog_Rate downscales the random trigger rate, it is a 32 bit register, split into two 16-bit registers. The rate can not be deduced directly from the register. Trig*_Prog_Delay gives defines the delay of a trigger in a normal mode. Actually the delay function has been removed in the firmware, so this register is not used now.

Table 2.1: Registers for Trigger0.

Register name	Address	Type	Width	Description
Trig0_OptionCode	0x00	R/W	2	Used for the selection of L0 output options
Trig0_Signature	0x01	R/W	7	Signature of Trigger0
Trig0_MessageHeader	0x02	R/W	8	Message Header of L0
Trig0_Prog_Rate_Low	0x03	R/W	16	The lowest 16 bits of Programmable Rate for Trigger0
Trig0_Prog_Rate_High	0x04	R/W	16	The highest 15 bits of Programmable Rate for Trigger0
Trig0_Prog_Delay	0x05	R/W	16	Programmable Delay for L0

Table 2.2: Registers for L1L.

Register name	Address	Type	Width	Description
Trig1L_OptionCode	0x06	R/W	2	Used for the selection of L1L output options
Trig1L_Signature	0x07	R/W	7	Signature of L1L
Trig1L_MessageHeader	0x08	R/W	8	Message Header of L1L
Trig1L_Prog_Rate_Low	0x09	R/W	16	The lowest 16 bits of Programmable Rate for L1L
Trig1L_Prog_Rate_High	0x0A	R/W	16	The highest 15 bits of Programmable Rate for L1L
Trig1L_Prog_Delay	0x0B	R/W	16	Programmable Delay for L1L

Table 2.3: Registers for L1M.

Register name	Address	Type	Width	Description
Trig1M_OptionCode	0x0C	R/W	2	Used for the selection of L1M output options
Trig1M_Signature	0x0D	R/W	7	Signature of L1M
Trig1M_MessageHeader	0x0E	R/W	8	Message Header of L1M
Trig1M_Prog_Rate_Low	0x0F	R/W	16	The Lowest 16bits of Programmable Rate for L1M
Trig1M_Prog_Rate_High	0x10	R/W	16	The highest 15bits of Programmable Rate for L1M
Trig1M_Prog_Delay	0x11	R/W	16	Programmable Delay for L1M

Table 2.4: Registers for L1H.

Register name	Address	Type	Width	Description
Trig1H_OptionCode	0x12	R/W	2	Used for the selection of L1H output options
Trig1H_Signature	0x13	R/W	7	Signature of L1H
Trig1H_MessageHeader	0x14	R/W	8	Message Header of L1H
Trig1H_Prog_Rate_Low	0x15	R/W	16	The Lowest 16 bits of Programmable Rate for L1H
Trig1H_Prog_Rate_High	0x16	R/W	16	The highest 15bits of Programmable Rate for L1H
Trig1H_Prog_Delay	0x17	R/W	16	Programmable Delay for L1H

2.2 General registers

The long table describes general registers in the TOR.

Therein, *Thre1*, *Thre2* and *Thre3* are three 14-bit threshold registers for L1L, L1M and L1H.

Mask_array is a 40-bit registers, masking off noisy/dead TRU inputs and implemented by three registers: two 16-bit registers and one 8-bit register.

Ctrl_reserve_r is a reserved register. Bit 0 is for testing L1 firmware. L1 is generated when the confirmed-L0 is received, when testing, a fake confirmed-L0 can be generated by asserting bit 0 of *Ctrl_reserve_r*. There are two clock domains, 40 MHz and 200 MHz. Two 32-bit registers (*dbg_clka, dbg_clkb*) record the number of clocks in a period (set by start and stop signals). The start and stop signals are asserted by bit 1 and bit 2 of *Ctrl_reserve_r* respectively. Bit 3 is a reset signal for registers *L0_confirmed_cnt* and *L0_inhibition_cnt*.

Four 32-bit Counters record the number of L0, L1L, L1M and L1H pulses in 1 second. Every counter is implemented by two 16-bit registers. For example, *Counter1* and *Counter 2* records L0 pulses, *Counter3* and *Counter 4* for L1L and so on.

Version register gives the version number of the firmware. For example, the firmware is 0x56.

M21_counter - M28_counter records L0 trigger pulses in one second of each L0 input channel (i.e. L0 from a TRU in Module 2) of TOR. *M31_counter - M38_counter* are for module 3, *M31_counter - M38_counter* are for module 4. The map is given in the following section.

Error_clk register indicate the Digital Clock Manager is locked or not. Only bit 0 is used, if the value is 0, the clock is locked, if not, the clock is not locked.

M21_cnt_in_L1L - M28_cnt_in_L1L, *M31_cnt_in_L1L - M38_cnt_in_L1L*, and *M41_cnt_in_L1L - M48_cnt_in_L1L* record L1L trigger pulses in one second of each L1L input channel of TOR.

M21_cnt_in_L1M - M28_cnt_in_L1M, *M31_cnt_in_L1M - M38_cnt_in_L1M* and *M41_cnt_in_L1M - M48_cnt_in_L1M* record L1M trigger pulses in one second of each L1M input channel of TOR.

M21_cnt_in_L1H - M28_cnt_in_L1H, *M31_cnt_in_L1H - M38_cnt_in_L1H* and *M41_cnt_in_L1H - M48_cnt_in_L1H* record L1H trigger pulses in one second of each L1H input channel of TOR.

Pulse_cnt1 records the number of L0 triggers with a length of 50 ns in a run. *Pulse_cnt2* and *pulse_cnt3* record the number of L0 triggers with lengths of 100 ns and 150 ns respectively. They are not implemented yet.

Pattern_L configures the delay in *SRL_16* for basic L1 generation.

Read_out_mask is a 32-bit register, setting inhibit time for L0. When confirmed-L0 is received, the inhibit function is active, the time period is set by this register. For example, if

1000 μs is needed, the value to be set in the register is 1000μs/25ns = 0x9C40.

L0_confirmed_cnt records the number of confirmed-L0, this is only for debug. *L0_inhibition_cnt* records number of clock cycles during inhibit time, this register is only for debug. Both two registers are reset by bit 4 of *Ctrl_reserve_r*.

Table 2.5: General registers.

Register name	Address	Type	Width	Description
Thre1	0x18	R/W	14	Threshold 1 for L1L
Thre2	0x19	R/W	14	Threshold 1 for L1M
Thre3	0x1a	R/W	14	Threshold 1 for L1H
Mask_array43	0x1b	R/W	16	L0 trigger Mask for module 3 and 4(high 8bits for 3)
Mask_array21	0x1c	R/W	16	L0 trigger Mask for module 1 and 2(high 8 bits for 1)
Mask_array0	0x1d	R/W	8	L0 trigger Mask for module 0
Ctrl_reserve_r	0x1e	R/W	16	Reserved control reg bit 0 is used for L1 test. Bit 1, bit 2 are used for clk_check. Bit 3 not used. Bit 4 reset for registers L0_confirmed_cnt and L0_inhibition_cnt.
Counter1	0x1f	R	16	The lowest 16 bits for L0 counter
Counter2	0x20	R	16	The highest 16 bits for L0 counter
Counter3	0x21	R	16	The lowest 16 bits for L1L counter
Counter4	0x22	R	16	The highest 16 bits for L1L counter
Counter5	0x23	R	16	The lowest 16 bits for L1M counter
Counter6	0x24	R	16	The highest 16 bits for L1M counter
Counter7	0x25	R	16	The lowest 16 bits for L1H counter
Counter8	0x26	R	16	The highest 16 bits for L1H counter
version	0x27	R	16	The version of the firmware
M41_counter	0x28	R	16	The input trigger counter of M41
M42_counter	0x29	R	16	The input trigger counter of M42
M43_counter	0x2a	R	16	The input trigger counter of M43
M44_counter	0x2b	R	16	The input trigger counter of M44
M45_counter	0x2c	R	16	The input trigger counter of M45
M46_counter	0x2d	R	16	The input trigger counter of M46

Register name	Address	Type	Width	Description
M47_counter	0x2e	R	16	The input trigger counter of M47
M48_counter	0x2f	R	16	The input trigger counter of M48
M31_counter	0x30	R	16	The input trigger counter of M31
M32_counter	0x31	R	16	The input trigger counter of M32
M33_counter	0x32	R	16	The input trigger counter of M33
M34_counter	0x33	R	16	The input trigger counter of M34
M35_counter	0x34	R	16	The input trigger counter of M35
M36_counter	0x35	R	16	The input trigger counter of M36
M37_counter	0x36	R	16	The input trigger counter of M37
M38_counter	0x37	R	16	The input trigger counter of M38
M21_counter	0x38	R	16	The input trigger counter of M21
M22_counter	0x39	R	16	The input trigger counter of M22
M23_counter	0x3a	R	16	The input trigger counter of M23
M24_counter	0x3b	R	16	The input trigger counter of M24
M25_counter	0x3c	R	16	The input trigger counter of M25
M26_counter	0x3d	R	16	The input trigger counter of M26
M27_counter	0x3e	R	16	The input trigger counter of M27
M28_counter	0x3f	R	16	The input trigger counter of M28
dbg_rdout4_L	0x40	R	16	not used
dbg_rdout4_H	0x41	R 16		not used
dbg_cmp_dout	0x42	R	16	not used
dbg_mlc_dout	0x43	R	16	not used
Scl_data_dout	0x44	R	16	not used
nfw_out	0x45	R	16	not used
dbg_clkb_L	0x46	R	16	The lowest 16 bits for clkb counter
dbg_clkb_H	0x47	R	16	The highest 16 bits for clkb counter
dbg_clka_L	0x48	R	16	The lowest 16 bits for clka counter
dbg_clka_H	0x49	R	16	The highest 16 bits for clka counter
error_clk	0x4A	R	16	The error register for CLK DCM
M41_cnt_in_L1L	0x4B	R	16	The input trigger counter of L1L for M41
M42_cnt_in_L1L	0x4C	R	16	The input trigger counter of L1L for M42
M43_cnt_in_L1L	0x4D	R	16	The input trigger counter of L1L for M43

Register name	Address	Type	Width	Description
M44_cnt_in_L1L	0x4E	R	16	The input trigger counter of L1L for M44
M45_cnt_in_L1L	0x4F	R	16	The input trigger counter of L1L for M45
M46_cnt_in_L1L	0x50	R	16	The input trigger counter of L1L for M46
M47_cnt_in_L1L	0x51	R	16	The input trigger counter of L1L for M47
M48_cnt_in_L1L	0x52	R	16	The input trigger counter of L1L for M48
M31_cnt_in_L1L	0x53	R	16	The input trigger counter of L1L for M31
M32_cnt_in_L1L	0x54	R	16	The input trigger counter of L1L for M32
M33_cnt_in_L1L	0x55	R	16	The input trigger counter of L1L for M33
M34_cnt_in_L1L	0x56	R	16	The input trigger counter of L1L for M34
M35_cnt_in_L1L	0x57	R	16	The input trigger counter of L1L for M35
M36_cnt_in_L1L	0x58	R	16	The input trigger counter of L1L for M36
M37_cnt_in_L1L	0x59	R	16	The input trigger counter of L1L for M37
M38_cnt_in_L1L	0x5A	R	16	The input trigger counter of L1L for M38
M21_cnt_in_L1L	0x5B	R	16	The input trigger counter of L1L for M21
M32_cnt_in_L1L	0x5C	R	16	The input trigger counter of L1L for M22
M23_cnt_in_L1L	0x5D	R	16	The input trigger counter of L1L for M23
M24_cnt_in_L1L	0x5E	R	16	The input trigger counter of L1L for M24
M25_cnt_in_L1L	0x5F	R	16	The input trigger counter of L1L for M25

Register name	Address	Type	Width	Description
M26_cnt_in_L1L	0x60	R	16	The input trigger counter of L1L for M26
M27_cnt_in_L1L	0x61	R	16	The input trigger counter of L1L for M27
M28_cnt_in_L1L	0x62	R	16	The input trigger counter of L1L for M28
pattern_L	0x63	W	16	The delay configuration of L1 generation
pattern_H	0x64	W	16	not used
dbg_dr_arr_L	0x65	R	16	not used
dbg_dr_arr_H	0x66	R	16	not used
M41_cnt_in_L1H	0x67	R	16	The input trigger counter of L1H for M41
M42_cnt_in_L1H	0x68	R	16	The input trigger counter of L1H for M42
M43_cnt_in_L1H	0x69	R	16	The input trigger counter of L1H for M43
M44_cnt_in_L1H	0x6A	R	16	The input trigger counter of L1H for M44
M45_cnt_in_L1H	0x6B	R	16	The input trigger counter of L1H for M45
M46_cnt_in_L1H	0x6C	R	16	The input trigger counter of L1H for M46
M47_cnt_in_L1H	0x6D	R	16	The input trigger counter of L1H for M47
M48_cnt_in_L1H	0x6E	R	16	The input trigger counter of L1H for M48
M31_cnt_in_L1H	0x6F	R	16	The input trigger counter of L1H for M31
M32_cnt_in_L1H	0x70	R	16	The input trigger counter of L1H for M32
M33_cnt_in_L1H	0x71	R	16	The input trigger counter of L1H for M33
M34_cnt_in_L1H	0x72	R	16	The input trigger counter of L1H for M34

Register name	Address	Type	Width	Description
M35_cnt_in_L1H	0x73	R	16	The input trigger counter of L1H for M35
M36_cnt_in_L1H	0x74	R	16	The input trigger counter of L1H for M36
M37_cnt_in_L1H	0x75	R	16	The input trigger counter of L1H for M37
M38_cnt_in_L1H	0x76	R	16	The input trigger counter of L1H for M38
M21_cnt_in_L1H	0x77	R	16	The input trigger counter of L1H for M21
M22_cnt_in_L1H	0x78	R	16	The input trigger counter of L1H for M22
M23_cnt_in_L1H	0x79	R	16	The input trigger counter of L1H for M23
M24_cnt_in_L1H	0x7A	R	16	The input trigger counter of L1H for M24
M25_cnt_in_L1H	0x7B	R	16	The input trigger counter of L1H for M25
M26_cnt_in_L1H	0x7C	R	16	The input trigger counter of L1H for M26
M27_cnt_in_L1H	0x7D	R	16	The input trigger counter of L1H for M27
M28_cnt_in_L1H	0x7E	R	16	The input trigger counter of L1H for M28
pulse_cnt1_L	0x7f	R	16	The number (low 16 bit) of received 50 ns L0
pulse_cnt1_H	0x80	R	16	The number (high 16 bit) of received 50 ns L0
pulse_cnt2_L	0x81	R	16	The number (low 16 bit) of received 100 ns L0
pulse_cnt2_H	0x82	R	16	The number (high 16 bit) of received 100 ns L0
pulse_cnt3_L	0x83	R	16	The number (low 16 bit) of received 150 ns L0
pulse_cnt3_H	0x84	R	16	The number (high 16 bit) of received 150 ns L0

Register name	Address	Type	Width	Description
M41_cnt_in_L1M	0x97	R	16	The number of received dr signal for CH0 in M4
M42_cnt_in_L1M	0x98	R	16	The number of received dr signal for CH1 in M4
M43_cnt_in_L1M	0x99	R	16	The number of received dr signal for CH2 in M4
M44_cnt_in_L1M	0x9A	R	16	The number of received dr signal for CH3 in M4
M45_cnt_in_L1M	0x9B	R	16	The number of received dr signal for CH4 in M4
M46_cnt_in_L1M	0x9C	R	16	The number of received dr signal for CH5 in M4
M47_cnt_in_L1M	0x9D	R	16	The number of received dr signal for CH6 in M4
M48_cnt_in_L1M	0x9E	R	16	The number of received dr signal for CH7 in M4
M31_cnt_in_L1M	0x9F	R	16	The number of received dr signal for CH0 in M3
M32_cnt_in_L1M	0xA0	R	16	The number of received dr signal for CH1 in M3
M33_cnt_in_L1M	0xA1	R	16	The number of received dr signal for CH2 in M3
M34_cnt_in_L1M	0xA2	R	16	The number of received dr signal for CH3 in M3
M35_cnt_in_L1M	0xA3	R	16	The number of received dr signal for CH4 in M3
M36_cnt_in_L1M	0xA4	R	16	The number of received dr signal for CH5 in M3
M37_cnt_in_L1M	0xA5	R	16	The number of received dr signal for CH6 in M3
M38_cnt_in_L1M	0xA6	R	16	The number of received dr signal for CH7 in M3
M21_cnt_in_L1M	0xA7	R	16	The number of received dr signal for CH0 in M2
M22_cnt_in_L1M	0xA8	R	16	The number of received dr signal for CH1 in M2

Register name	Address	Type	Width	Description
M23_cnt_in_L1M	0xA9	R	16	The number of received dr signal for CH2 in M2
M24_cnt_in_L1M	0xAA	R	16	The number of received dr signal for CH3 in M2
M25_cnt_in_L1M	0xAB	R	16	The number of received dr signal for CH4 in M2
M26_cnt_in_L1M	0xAC	R	16	The number of received dr signal for CH5 in M2
M27_cnt_in_L1M	0xAD	R	16	The number of received dr signal for CH6 in M2
M28_cnt_in_L1M	0xAE	R	16	The number of received dr signal for CH7 in M2
Read_out_mask_L	0xAF	W/R	16	The lowest 16 bits for trigger mask.
Read_out_mask_H	0xB0	W/R	16	The highest 16 bits for trigger mask.
L0_confirmed_cnt	0xB1	R	16	The counter for L0_confirmed
L0_inhibition_cnt	0xB2	R	16	The counter for L0_inhibition

Chapter 3

The operation of TOR at P2

3.1 How to configure TOR?

1. First, log on alidcsdcb1572 by typing “tor” on the machine alidcsdcb075, the password is dcs.

2. Then type “ps” to check the tinserver is running, if not,run “./mnt/dcbrw/starttinserver” and check again.

3. Then check if the firmware is there.

   Type “rcu-sh r 0x27”, if the answer is 0x56,the FPGA has been programmed with firmware now. If you get “no target answer”, the FPGA doesn’t have firmware yet,you need to program it using the following command:

   “./program_tor tor_fpga2_300611.bit”

   Then check the firmware again.

4. Finally, 0x8340 should be written in the readout_mask_register 0xaf by type “rcu-sh w 0xaf 0x8340”. This is an inhibit time register, the value times 25 ns determines the inhibit time, during which no L0 trigger is sent to CTP.

5. Well, now you can initialize the TOR by “rcu-sh b /mnt/dcbrw/set_register.scr”. The default mode is normal mode, you can change it by writing the corresponding optioncode.

Now the TOR is ready for test.

3.2 Test the test mode in TOR

Each Trigger has 4 options: normal, toggling, random, and signature. To check the link between the CTP and TOR, you can select toggling and signature. Then how to set the mode?

For L0:
rcu-sh w 0x00 0x02 //Signature Mode
rcu-sh w 0x00 0x03 // Random
rcu-sh w 0x00 0x01 // Toggle
rcu-sh w 0x00 0x00 //Normal
For L1L:
rcu-sh w 0x06 0x02 //Signature Mode
rcu-sh w 0x06 0x03 // Random
rcu-sh w 0x06 0x01 // Toggle
rcu-sh w 0x06 0x00 // Normal
For L1M:
rcu-sh w 0x0c 0x02 //Signature Mode
rcu-sh w 0x0c 0x03 // Random
rcu-sh w 0x0c 0x01 // Toggle
rcu-sh w 0x0c 0x00 // Normal
For L1H:
rcu-sh w 0x12 0x02 //Signature Mode
rcu-sh w 0x12 0x03 // Random
rcu-sh w 0x12 0x01 // Toggle
rcu-sh w 0x12 0x00 // Normal

Note: The signature of L0 is 4, L1L is 5, L1M is 6 and L1H is 7.If you can't get correct signature from CTP screen, try to read signature register, if the answer is 0,then you need to initialize the TOR.

3.3 Test the trigger

3.3.1 Test L0 trigger

Actually the Mask_array43 and Mask_array21 have been set to 0xffff and 0xff respectively, which choose all the TRUs of Three modules.

Now, if you want to choose only one TRU, for instance PHOS-3-1,then use the following command:

rcu-sh w 0x1c 0x01
rcu-sh w 0x1b 0x00
you can read the trigger counter by "rcu-sh b /mnt/dcbrw/read_counter.scr".

The counter has 32 bits, register 0x1f records the lowest 16 bits, register 0x20 records the highest 16 bits. Remember the counter records the number of triggers in 1 seconds, and it

Table 3.1: L0 counter address, TRUs and corresponding bit for mask. This is for M2, the mask register is 0x1c.

counter address	TRUs	bit in mask register
0x38	84_B	0
0x39	87_B	1
0x3a	86_B	2
0x3b	87_B	3
0x3c	84_A	4
0x3d	87_A	5
0x3e	86_A	6
0x3f	85_A	7

refreshes every 1 seconds.

For every L0 input, there is a 16-bit counter for it, the counter records the number of L0 from TRU in 1 second. You can run "rcu-sh b read_counter_M2_in.scr" to check for Module 2. "rcu-sh b read_counter_M3_in.scr" to check Module 3 and "rcu-sh b read_counter_M4_in.scr" to check Module 4.

The L0 counter address, the corresponding TRUs and the bit you need to set to mask off (the corresponding bit is set to 0 to mask off) are described in Table 3.1 and Table 3.2.

A different test mode is implemented in the TRU, and it can be used to test the link between the TRU and the TOR. The test mode can be enabled by writing 0x000f to register 0x04 by "./TRU_write.sh A 04 0x000f"; the trigger rate can be adjusted by writing register 0x07. For example, if you need 1000 Hz, the value x, which should write to register 0x07, is $x = 40000000/1000 = 40000$.

For L0, L1L, L1M and L1H, each of them has its own test mode module.

3.3.2 Test L1 trigger

As mentioned above, the test modes in the TRU for the L1 links are also implemented. "./TRU_test_mode_set.sh 0xxxxx" can configure the TRU works in the test mode. It means: send programmable trigger rate to the TOR. On the TOR, read_counter_L1L.txt, read_counter_L1M.txt and read_counter_L1H.txt can be executed to check the trigger rate on the TOR side. read_counter_L1L.txt can be run by typing "rcu-sh b read_counter_L1L.txt" in directory "/mnt/dcbrw/". If you use 0x9c40 when configuring the test mode, the value of all the counters should be 0x3e8. If not, there is something wrong with the link.

Table 3.2: L0 counter address, TRUs and corresponding bit for mask. This is for M3 and M4, the mask register is 0x1b.

counter address	TRUs	bit in mask register
0x28	88_B	0
0x29	89_B	1
0x2a	90_B	2
0x2b	91_B	3
0x2c	88_A	4
0x2d	89_A	5
0x2e	90_A	6
0x2f	91_A	7
0x30	95_B	8
0x31	94_B	9
0x32	93_B	10
0x33	92_B	11
0x34	95_A	12
0x35	94_A	13
0x36	93_A	14
0x37	92_A	15

3.4 The Map between TRU and TOR at p2

From the front view of the TOR shown in Figure 3.1, 40 inputs of the TOR are allocated to the TRUs in Module 1, Module 2, Module 3, Module 4 and Module 5 from the left to the right.

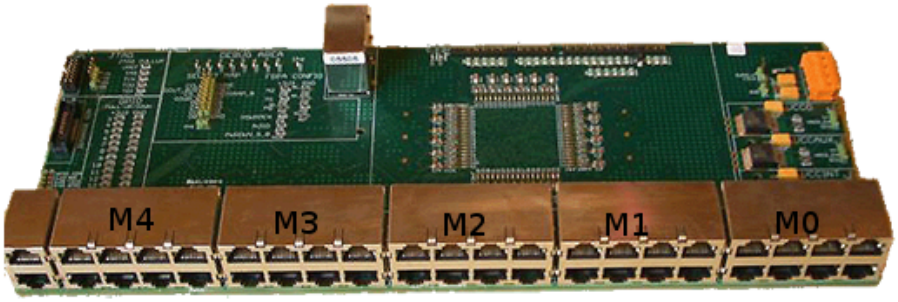

Figure 3.1: The TOR inputs allocation (Front view).

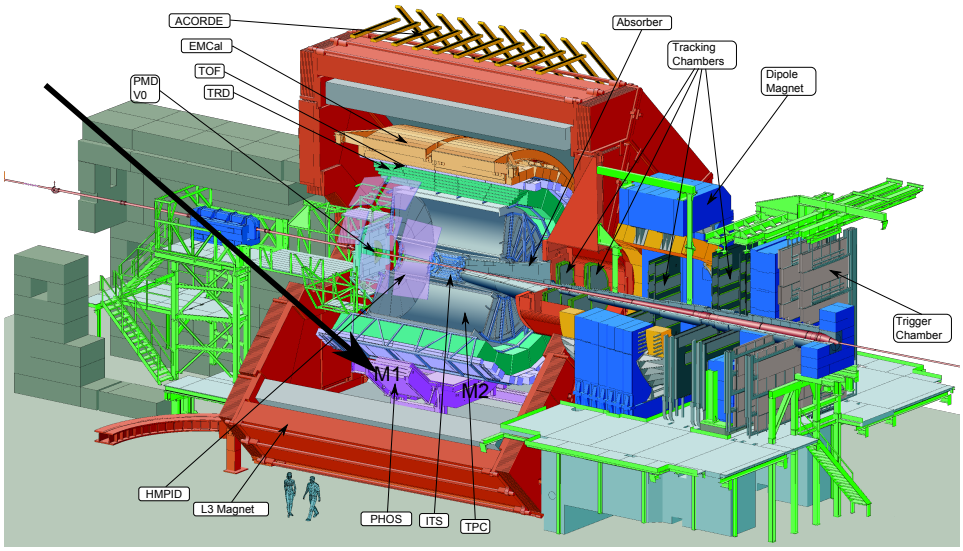

Figure 3.2: The PHOS modules location in the ALICE experiment .

The location of 5 Modules are shown in Figure 3.2. If you look from the outside of the door, pointed as the bold arrow shows in the figure, they are M0, M1, M2, M3, M4 from the left to the right. Table 3.3 describes the connections between TRUs and the TOR. Each module consists of 8 TRUs, connected to 8 inputs of the TOR. For example, 8 TRUs in Module 2 are collected to the left most 8 inputs of the TOR in Figure 3.1. The top 4 inputs out of 8 are allocated to TRU-1, TRU-3, TRU-5 and TRU-7; the bottom 4 inputs are for TRU-2, TRU-4, TRU-6 and TRU-8. Two TRUs in the same column are controlled by one DCS board when they are configured. For instance, TRU-1 and TRU-2 are configured by DCS board 84 in Module 2.

Table 3.3: The map between TRUs, TOR and DCS for configuring the TRUs at P2.

Module 0				Module 1				Module 2				Module 3				Module 4				
1	3	5	7	1	3	5	7	1	3	5	7	1	3	5	7	1	3	5	7	B
2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	A
								84	87	86	85	95	94	93	92	88	89	90	91	dcs

When the TRUs are to be programmed, different maps are used. Table 3.4 describes the map between TRUs and DCSs. If you want to program TRU-1 in Module 2, just log on 84, then use the erase1.xsvf and prog1.xsvf; if you want to program TRU-3 in this Module, log on 84, then use the erase2.xsvf and prog2.xsvf instead.

Table 3.4: The map between TRUs and DCS at P2 for programming TRUs.

Module 2				Module 3				Module 4				
1 84-1	3 84-2	5 86-1	7 86-2	1 95-1	3 95-2	5 93-1	7 93-2	1 88-1	3 88-2	5 90-1	7 90-2	B
2 87-2	4 87-1	6 85-2	8 85-1	2 94-2	4 94-1	6 92-2	8 92-1	2 89-2	4 89-1	6 91-2	8 91-1	A

The inputs of the TOR are at high level by default. When some TRUs are not available, the corresponding inputs are '1', which disturbs the trigger processing in the TOR. If only some of them take part in the trigger generation for some reason, a mask is used to mask unavailable TRUs off. The following Table 3.5 gives the map for the mask. M21_0 means the bit 0 of Mask_array21, which corresponds to the TRU-1 in Module 2.

Table 3.5: The mask for the TOR inputs.

Module2				Module3				Module4				
1 M21-0	3 M21-1	5 M21-2	7 M21-3	1 M43-8	3 M43-9	5 M43-10	7 M43-11	1 M43-0	3 M43-1	5 M43-2	7 M43-3	B
2 M21-4	4 M21-5	6 M21-6	8 M21-7	2 M43-12	4 M43-13	6 M43-14	8 M43-15	2 M43-4	4 M43-5	6 M43-6	8 M43-7	A

On the TOR side, there is a trigger input counter for each TRU, the counter names and the corresponding TRU are described in Table 3.6. Only Module 4 is shown here for example.

Table 3.6: The trigger counter registers in TOR for the TRUs.

Module4				
1 M41_counter	3 M42_counter	5 M43_counter	7 M44_counter	B
2 M45_counter	4 M46_counter	6 M47_counter	8 M48_counter	A

Bibliography

[1] A. Jusko and O. Villalobos-Baillie, *Requirements for Detectors Supplying Trigger inputs*, 2005, http://www.ep.ph.bham.ac.uk/user/ovb/Automatic_timing.doc. 8

[2] A. trigger group, *Trigger Output Logic-Hardware Guide for Front-end Designers*, 2007, http://epweb2.ph.bham.ac.uk/user/krivda/alice/ctp/ctp.htm. 8, 9

[3] ALICE-CTP-group, *TIN proxy*, http://epweb2.ph.bham.ac.uk/user/jusko/ctpinputs/index.html. 8