TRU instructions

Dong & Lijiao

Red color stands for the completed command you should type.
Firmware version:V2033
Erase file :erasefirst.xsvf,erasesecond.xsvf.
Program file: v2033_1.xsvf, v2033_2.xsvf.

Then you can start these procedures:
Generally, you don't need to program the TRUs, after the TRU are powered on, the firmware would be loaded automatically.
Before you configure the TRU, make sure that the low voltage and FEE are on.
Then TRU needs to be configured firstly.
1--Log on DCS
2-- go to /mnt/dcbrw/tru-dcs-share/tru_script
3-- type ./TRU_initial.sh the threshold is set to 0x0100 after initialization, you can change it by ./TRU_threshold_A.sh 0x0050 and ./TRU_threshold_B.sh 0x0050. Replace 0x0050 with the threshold you want
4—After initialization, the screen looks like this:
A:0x8002:0x2033
0x8003:0
0x8002 0x4
0x8003:0
B:0x8002:0x102033
0x8003:0
0x8002 0x100004
0x8003:0

After the initialization, the threshold is set to 0x100, the pedestal and mask have been set. If the LED is off, the trigger rate should be 1HZ or 2 HZ.
5 – If the Fakealtro is supposed to be readout.
Two modes can be selected by writing register 0x7b:
'0'—readout Fakealtro data and trigger information 112*10+91 L0 trigger
'1'---just readout trigger information 91 L0 trigger.

How to Write/Read register:

If you want to write or read register, the feeserver should be closed firstly.
If you want to write Data A to register B:
Run the following command:
rcu-sh w 0x8005 B
rcu-sh w 0x8006 A
rcu-sh w 0x8010 0x00
If you want to read a register A:
rcu-sh w 0x8005 A
rcu-sh w 0x8010 0x00
rcu-sh r 0x8002
The data of the register would show in 0x8002.

Actually, there is script for the write/read register, go to /mnt/dcbrw/tru-dcs-share/tru_script

If you want write 0x0000 to address 0x70 in branch A:
./TRU_write.sh A 70 0x0000
If you want read address 0x70 in branch A:
./TRU_read.sh A 70

How to program the TRU

Concerning how to program the TRU(Don't do this generally unless you really need):
1: kill the feeserver, switch off HV of three modules, switch off FEE cards
2:Log on DCS for RCU:/home/phs 84 (--84~95 for three module)
3.Check the TRU adapt card
./mnt/dcbrw/tru-dcs-share/jtagop c
If there are 4 devices, the adapt cards are good, if not, then TRU needs to be switched on by the following command:
rcu-sh w 0x5100 0x10001
then run ./mnt/dcbrw/tru-dcs-share/jtagop c again. There should be 4 devices now. If not, switch on FEE cards by ./mnt/dcbrw/tru-dcs-share/feestart.sh and switch off them again ./mnt/dcbrw/tru-dcs-share/feestop.sh and try ./mnt/dcbrw/tru-dcs-share/jtagop c again.
4.Erase two TRUs controlled by this RCU.
./mnt/dcbrw/tru-dcs-share/playxsvf –t /dev/jtag /mnt/dcbrw/tru-dcs-share/tru_fw/ erasefirst.xsvf
Be patient, it takes some time, maybe 1 minute
./mnt/dcbrw/tru-dcs-share/playxsvf –t /dev/jtag /mnt/dcbrw/tru-dcs-share/tru_fw/ erasesecond.xsvf
Be patient, it takes some time, maybe 1 minute

4.Switch off TRU and Switch on them again;
rcu-sh r 0x5100 0x0
rcu-sh w 0x5100 0x10001

5.Program the two TRUs
./mnt/dcbrw/tru-dcs-share/playxsvf –t /dev/jtag /mnt/dcbrw/tru-dcs-share/ tru_fw/ v2033_1.xsvf
Be patient, it takes some time, maybe 10 minutes

6. Power cycle for TRU, do the same as 4.
./mnt/dcbrw/tru-dcs-share/playxsvf –t /dev/jtag /mnt/dcbrw/tru-dcs-share/ tru_fw/ v2033_2.xsvf
Be patient, it takes some time, maybe 10 minutes
Note: the new firmware you program will work after you do power cycle of the whole module.
On DCS 88, there is only one TRU available, so eraseone.xsvf and v2033_single are for it.

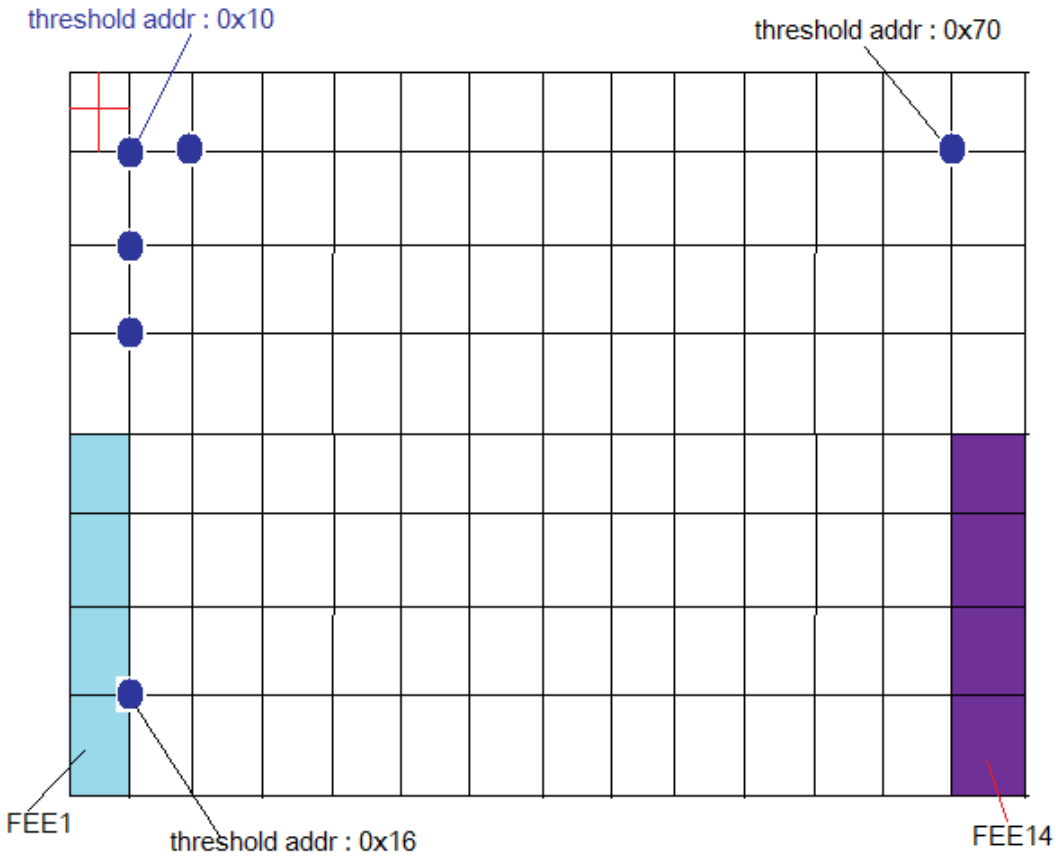
PHOS TRU register specification(V2033)
Draft by Dong Wang
2011.1.18

Caution : you need use Slow controller bus to read and write!

Register name	Addr	Type	With	Description
Version_Number	0x00	R	16 bit	Version Number is 0x1101
ADC_Set_Addr	0x01	R/W	4 bit	ADC Chip select signal. 0 : select ADS5270 IC0 1 : select ADS5270 IC1 … 13 : select ADS5270 IC13 15 : Broadcast to all ADS5270.
ADC_Set_Mode	0x02	R/W	5bit	Command input. 5'b00001 : Initial ADS5270 5'b00010 : Set ADS5270 to Normal ADC mode 5'b00100 : Set ADS5270 to deskew mode 5'b01000 : Set ADS5270 to sync mode 5'b10000 : Set ADS5270 to custom mode
Phase_Shift	0x03	R/W	10 bit	Phase_shift for ADC sampling clock
L0,L1-L,L1-M,L1_H, test mode control	0x04	R/W	1bit	L0 : controlled by LSB (bit 0) L1_L : bit 1 L1_M : bit 2 L1_H : bit 3 '0' normal L0 output '1' L0 output controlled by frequency
Trigger_counter_reset	0x05	R/W	1bit	'1' reset '0' nomal
Trigger_counter	0x06	R	16bit	Trigger counter
L0_freq_counter	0x07	R/W	16bit	L0 trigger test mode, output controlled by 40MHZ/counter

Mask_Channel	0x08 – 0x0E	R/W	16bit	Channel Mask for 112 channels. From left to right (FEE1 to FEE 14 for Branch A. FEE14 to FEE1 for Branch B). Branch A: 0x08 : FEE 1- FEE 2 0x09 : FEE 3-FEE 4 . . 0x0e : FEE13-FEE14 ‘0’ not mask normal mode ‘1’ mask mode
Threshold (Branch A map)	0x10 – 0x1D	R/W	16bit	0x10 – 0x16 : 4x4 space sum threshold between FEE1 and FEE2 0x17 – 0x1D : 4x4 space sum threshold between FEE2 and FEE3
	0x20 – 0x2D	R/W	16bit	0x20 – 0x26 : 4x4 space sum threshold between FEE3 and FEE4 0x27 – 0x2D : 4x4 space sum threshold between FEE4 and FEE5
	0x30 – 0x3D	R/W	16bit	0x30 – 0x36 : 4x4 space sum threshold between FEE5 and FEE6 0x37 – 0x3D : 4x4 space sum threshold between FEE6 and FEE7
	0x40 – 0x4D	R/W	16bit	0x40 – 0x46 : 4x4 space sum threshold between FEE7and FEE8 0x47 – 0x4D : 4x4 space sum threshold between FEE8 and FEE9
	0x50 – 0x5D	R/W	16bit	0x50 – 0x56 : 4x4 space sum threshold between FEE9 and FEE10 0x57 – 0x5D : 4x4 space sum threshold between FEE10and FEE11
	0x60 – 0x6D	R/W	16bit	0x60 – 0x66 : 4x4 space sum threshold between FEE11 and FEE12 0x67 – 0x6D : 4x4 space sum threshold between FEE12 and FEE13

	0x70 – 0x76	R/W	16bit	0x70 – 0x76 : 4x4 space sum threshold between FEE13 and FEE14
L1_L_freq_counter	0x78	R/W	16bit	L1_L trigger test mode, output controlled by 40MHZ/counter
L1_M_freq_counter	0x79	R/W	16bit	L1_M trigger test mode, output controlled by 40MHZ/counter
L1_H_freq_counter	0x7a	R/W	16bit	L1_M trigger test mode, output controlled by 40MHZ/counter
Fakealtro mode	0x7b	R/W	1bit	'0'—readout data and trigger information 112*10+91 L0 trigger '1'---just readout trigger information 91 L0 trigger
Pedestal_register	0x80-0xEF	R/W	12bit	112 channel pedestal set
L1-low	0xf0	R/W	16bit	Gloal threshold register for L1 low level
L1-meddle	0xf1	R/W	16bit	Gloal threshold register for L1 middle level
L1-high	0xf2	R/W	16bit	Gloal threshold register for L1 high level
ADC Pattern	0xf5	R/W	12bit	ADC pattern register
Pattern compare with pedestal counter reset	0xf6	R/W	1bit	Rest the counter for Pattern compare
Pattern compare with pedestal counter	0xf7	R	16bit	The counter of pattern compared with pedestal. If they are not equal the counter will add one .
Trigger place	0xf8 0xf9 0xfa 0xfb 0xfc 0xfd	R/W	16bit	0-15 16-31 32-47 48-63 64-79 80-95 The bit 91 is the global L0 output The bit 95 is the select bit: '0' normal; '1' test mode



Phase_Shift controlling interface

One suggested control scheme for controlling the phase of the output clock via DCS, the scheme description is showed below :

Addr	Acc	Width	Register Name	Format
0x03	R/W	10	Phase_Shift	[9:4] : 6 bits, *ps_step*, define the phase shift steps (R/W) [3] : *psincdec* (R/W) [2:0] : *dcm_status_out* (Read only)

It uses only one register for simplifying the operation and saving register space. When someone writes this register, this operation will redefine *ps_step* and *psincdec*, and activate *dcm_ps_cmd*. When someone reads this register, it will return the current configuration and status of the Tunable Phase Shift Module.

NOTE: The address here is for write Branch A. If you want to write Branch B, then the address is 0x1xxx, if you read Branch A, the address is 0x4xxx, if you read Branch B,the address is 0x5xxx.
For example: the address given here for ADC_Set_Mode is 0x02,then
Write ADC_Set_Mode in branch A address : 0x0002
Write ADC_Set_Mode in branch B address : 0x1002
Read ADC_Set_Mode in branch A address: 0x4002
Read ADC_Set_Mode in branch B address: 0x5002
Complicated, right? But it works this way, you have to comply.

Instructions for TOR
Lijiao

1.configure TOR

1--Firstly, log on 72, the password is dcs.
2--Then type ps to check the tinserver is running, if not,run ./mnt/dcbrw/startserver and check again.
3--Then check if the firmware is there.
Type rcu-sh r 0x27,
If the answer is 0x49,the FPGA has firmware now.
If you get no target answer,the FPGA doesn't have firmware yet,you need program it according the following command:
rcu-sh w 0xbf01 0x01
cat /mnt/dcbrw/tor_fpga2_040211.bit > /dev/virtex1
rcu-sh w 0xbf01 0x00
Then check the firmware again.
4--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.

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 TOR.

3. test the trigger

A --Test L0:

Actually the Mask_array12 and Mask_array34 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

Since electronics in 95A,88B,93A,93B are turned off, register 0x1b should be written 0xabfe by manual, 0x1c is still 0xff.

you can read the trigger counter by rcu-sh b /mnt/dcbrw/read_counter.scr

The counter has 32 bits, register 0x1f records the low 16 bits, register 0x20 records high 16 bits. Remember the counter records the number of trigger in 1 seconds, and it 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.

A testmode in TRU is implemented, which can be used to test the link between TRU and TOR. The testmode 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, there are four separated testmode module for them. The registers for L1 can be found in TRU register specification.

B --Test L1

The testmode in TRU for L1 link between TRU and TOR , and the testmode in TOR for the link to the CTP are implemented.

./TRU_test_mode_set.sh 0xxxxx can configure TRU works in Testmode, it means send programmable trigger rate to TOR. In TOR, read_counter_L1L.txt, read_counter_L1M.txt and read_counter_L1H.txt can be read out to check the trigger rate on TOR side.

read_counter_L1L.txt can be run by type rcu-sh b read_counter_L1L.txt in directory /mnt/dcbrw/. If you write 0x9c40 when configure testmode, the value of all the counters should be 0x3e8, if not, there is something wrong with the link.

4. List of the registers

Register name	Address	Type [1]	Description
Trig0_OptionCode	0x00	R/W	Used for the selection of trigger0 output options
Trig0_Signature	0x01	R/W	Signature of Trigger0
Trig0_MessageHeader	0x02	R/W	Message Header of Trigger0
Trig0_Prog_Rate_Low	0x03	R/W	Low 16bits of Programmable Rate for Trigger0
Trig0_Prog_Rate_High	0x04	R/W	High15bits of Programmable Rate for Trigger0
Trig0_Prog_Delay	0x05	R/W	Programmable Delay for Trigger0

Table 3-1:Registers for Trigger0

[1] Legend: W=write, R=read, T= write trigger (not physical registers)

Register name	Address	Type [2]	Description
Trig1L_OptionCode	0x06	R/W	Used for the selection of trigger1L output options
Trig1L_Signature	0x07	R/W	Signature of Trigger1L
Trig1L_MessageHeader	0x08	R/W	Message Header of Trigger1L
Trig1L_Prog_Rate_Low	0x09	R/W	Low 16bits of Programmable Rate for Trigger1L
Trig1L_Prog_Rate_High	0x0A	R/W	High15bits of Programmable Rate for Trigger1L
Trig1L_Prog_Delay	0x0B	R/W	Programmable Delay for Trigger1L

Table 3-2:Registers for Trigger1L

Register name	Address	Type [3]	Description
Trig1M_OptionCode	0x0C	R/W	Used for the selection of trigger1M output options
Trig1M_Signature	0x0D	R/W	Signature of Trigger1M
Trig1M_MessageHeader	0x0E	R/W	Message Header of Trigger1M
Trig1M_Prog_Rate_Low	0x0F	R/W	Low 16bits of Programmable Rate for Trigger1M
Trig1M_Prog_Rate_High	0x10	R/W	High15bits of Programmable Rate for Trigger1M
Trig1M_Prog_Delay	0x11	R/W	Programmable Delay for Trigger1M

Table 3-3:Registers for Trigger1M

Register name	Address	Type [4]	Description
Trig1H_OptionCode	0x12	R/W	Used for the selection of trigger1H output options
Trig1H_Signature	0x13	R/W	Signature of Trigger1H
Trig1H_MessageHeader	0x14	R/W	Message Header of Trigger1H
Trig1H_Prog_Rate_Low	0x15	R/W	Low 16bits of Programmable Rate for Trigger1H
Trig1H_Prog_Rate_High	0x16	R/W	High15bits of Programmable Rate for Trigger1H
Trig1H_Prog_Delay	0x17	R/W	Programmable Delay for Trigger1H

Table 3-4:Registers for Trigger1H

Register name	Address	Type [5]	Description
Thre1	0x18	R/W	Threshold 1 for L1L
Thre2	0x19	R/W	Threshold 1 for L1M
Thre3	0x1a	R/W	Threshold 1 for L1H
Mask_array12	0x1b	R/W	L0 trigger Mask for module 2 and 1(high 8bits for 2)
Mask_array34	0x1c	R/W	L0 trigger Mask for module 4 and 3(high 8bits for 4)
Mask_array5	0x1d	R/W	L0 trigger Mask for module 5
Ctrl_reserve_r	0x1e	R/W	Reserved control reg,bit 0 is used for L1 test. Bit1,bit 2 are used for clk_check,bit 3 is for trig_cnt,bit 4 is for communication test.

Table 3-5:General registers

Register name	Address	Type [6]	Description

[2] Legend: W=write, R=read, T= write trigger (not physical registers)

[3] Legend: W=write, R=read, T= write trigger (not physical registers)

[4] Legend: W=write, R=read, T= write trigger (not physical registers)

[5] Legend: W=write, R=read, T= write trigger (not physical registers)

[6] Legend: W=write, R=read, T= write trigger (not physical registers)

Counter1	0x1f	R	The low 16 bits for L0 counter
Counter2	0x20	R	The high 16 bits for L0 counter
Counter3	0x21	R	The low 16 bits for L1L counter
Counter4	0x22	R	The high 16 bits for L1L counter
Counter5	0x23	R	The low 16 bits for L1M counter
Counter6	0x24	R	The high 16 bits for L1M counter
Counter7	0x25	R	The low 16 bits for L1H counter
Counter8	0x26	R	The high 16 bits for L1H counter
version	0x27	R	The version of the firmware
M11_counter	0x28	R	The input trigger counter of M11
M12_counter	0x29	R	The input trigger counter of M12
M13_counter	0x2a	R	The input trigger counter of M13
M14_counter	0x2b	R	The input trigger counter of M14
M15_counter	0x2c	R	The input trigger counter of M15
M16_counter	0x2d	R	The input trigger counter of M16
M17_counter	0x2e	R	The input trigger counter of M17
M18_counter	0x2f	R	The input trigger counter of M18
M21_counter	0x30	R	The input trigger counter of M21
M22_counter	0x31	R	The input trigger counter of M22
M23_counter	0x32	R	The input trigger counter of M23
M24_counter	0x33	R	The input trigger counter of M24
M25_counter	0x34	R	The input trigger counter of M25
M26_counter	0x35	R	The input trigger counter of M26
M27_counter	0x36	R	The input trigger counter of M27
M28_counter	0x37	R	The input trigger counter of M28
M31_counter	0x38	R	The input trigger counter of M31
M32_counter	0x39	R	The input trigger counter of M32
M33_counter	0x3a	R	The input trigger counter of M33
M34_counter	0x3b	R	The input trigger counter of M34
M35_counter	0x3c	R	The input trigger counter of M35
M36_counter	0x3d	R	The input trigger counter of M36
M37_counter	0x3e	R	The input trigger counter of M37
M38_counter	0x3f	R	The input trigger counter of M38
dbg_rdout4_L	0x40	R	
dbg_rdout4_H	0x41	R	
dbg_cmp_dout	0x42	R	
dbg_mlc_dout	0x43	R	
Scl_data_dout	0x44	R	
nfw_out	0x45	R	

Table 3-6: Read registers

Register name	Address	Type [7]	Description
dbg_clkb_L	0x46	R	The low 16 bits for clkb counter
dbg_clkb_H	0x47	R	The high 16 bits for clkb counter
dbg_clka_L	0x48	R	The low 16 bits for clka counter
dbg_clka_H	0x49	R	The high 16 bits for clka counter
error_clk	0x4A	R	The error register for CLK DCM
M11_cnt_in_L1L	0x4B	R	The input trigger counter of L1L for M11
M12_cnt_in_L1L	0x4C	R	The input trigger counter of L1L for M12
M13_cnt_in_L1L	0x4D	R	The input trigger counter of L1L for M13
M14_cnt_in_L1L	0x4E	R	The input trigger counter of L1L for M14
M15_cnt_in_L1L	0x4F	R	The input trigger counter of L1L for M15
M16_cnt_in_L1L	0x50	R	The input trigger counter of L1L for M16

[7] Legend: W=write, R=read, T= write trigger (not physical registers)

M17_cnt_in_L1L	0x51	R	The input trigger counter of L1L for M17
M18_cnt_in_L1L	0x52	R	The input trigger counter of L1L for M18
M21_cnt_in_L1L	0x53	R	The input trigger counter of L1L for M21
M22_cnt_in_L1L	0x54	R	The input trigger counter of L1L for M22
M23_cnt_in_L1L	0x55	R	The input trigger counter of L1L for M23
M24_cnt_in_L1L	0x56	R	The input trigger counter of L1L for M24
M25_cnt_in_L1L	0x57	R	The input trigger counter of L1L for M25
M26_cnt_in_L1L	0x58	R	The input trigger counter of L1L for M26
M27_cnt_in_L1L	0x59	R	The input trigger counter of L1L for M27
M28_cnt_in_L1L	0x5A	R	The input trigger counter of L1L for M28
M31_cnt_in_L1L	0x5B	R	The input trigger counter of L1L for M31
M32_cnt_in_L1L	0x5C	R	The input trigger counter of L1L for M32
M33_cnt_in_L1L	0x5D	R	The input trigger counter of L1L for M33
M34_cnt_in_L1L	0x5E	R	The input trigger counter of L1L for M34
M35_cnt_in_L1L	0x5F	R	The input trigger counter of L1L for M35
M36_cnt_in_L1L	0x60	R	The input trigger counter of L1L for M36
M37_cnt_in_L1L	0x61	R	The input trigger counter of L1L for M37
M38_cnt_in_L1L	0x62	R	The input trigger counter of L1L for M38
pattern_L	0x63	W	The delay configuration of L1 generation
pattern_H	0x64	W	
dbg_dr_arr_L	0x65	R	
dbg_dr_arr_H	0x66	R	
M11_cnt_in_L1H	0x67	R	The input trigger counter of L1H for M11
M12_cnt_in_L1H	0x68	R	The input trigger counter of L1H for M12
M13_cnt_in_L1H	0x69	R	The input trigger counter of L1H for M13
M14_cnt_in_L1H	0x6A	R	The input trigger counter of L1H for M14
M15_cnt_in_L1H	0x6B	R	The input trigger counter of L1H for M15
M16_cnt_in_L1H	0x6C	R	The input trigger counter of L1H for M16
M17_cnt_in_L1H	0x6D	R	The input trigger counter of L1H for M17
M18_cnt_in_L1H	0x6E	R	The input trigger counter of L1H for M18
M21_cnt_in_L1H	0x6F	R	The input trigger counter of L1H for M21
M22_cnt_in_L1H	0x70	R	The input trigger counter of L1H for M22
M23_cnt_in_L1H	0x71	R	The input trigger counter of L1H for M23
M24_cnt_in_L1H	0x72	R	The input trigger counter of L1H for M24
M25_cnt_in_L1H	0x73	R	The input trigger counter of L1H for M25
M26_cnt_in_L1H	0x74	R	The input trigger counter of L1H for M26
M27_cnt_in_L1H	0x75	R	The input trigger counter of L1H for M27
M28_cnt_in_L1H	0x76	R	The input trigger counter of L1H for M28
M31_cnt_in_L1H	0x77	R	The input trigger counter of L1H for M31
M32_cnt_in_L1H	0x78	R	The input trigger counter of L1H for M32
M33_cnt_in_L1H	0x79	R	The input trigger counter of L1H for M33
M34_cnt_in_L1H	0x7A	R	The input trigger counter of L1H for M34
M35_cnt_in_L1H	0x7B	R	The input trigger counter of L1H for M35
M36_cnt_in_L1H	0x7C	R	The input trigger counter of L1H for M36
M37_cnt_in_L1H	0x7D	R	The input trigger counter of L1H for M37
M38_cnt_in_L1H	0x7E	R	The input trigger counter of L1H for M38
Dist_L0_confirmL0	0x7f	R	The distance of L0 and confirmed L0(in clock cycles)
M11_cnt_in_L1M	0x97	R	The number of received dr signal for CH0 in M1
M12_cnt_in_L1M	0x98	R	The number of received dr signal for CH1 in M1
M13_cnt_in_L1M	0x99	R	The number of received dr signal for CH2 in M1
M14_cnt_in_L1M	0x9A	R	The number of received dr signal for CH3 in M1
M15_cnt_in_L1M	0x9B	R	The number of received dr signal for CH4 in M1
M16_cnt_in_L1M	0x9C	R	The number of received dr signal for CH5 in M1
M17_cnt_in_L1M	0x9D	R	The number of received dr signal for CH6 in M1
M18_cnt_in_L1M	0x9E	R	The number of received dr signal for CH7 in M1
M21_cnt_in_L1M	0x9F	R	The number of received dr signal for CH0 in M2
M22_cnt_in_L1M	0xA0	R	The number of received dr signal for CH1 in M2

M23_cnt_in_L1M	0xA1	R	The number of received dr signal for CH2 in M2
M24_cnt_in_L1M	0xA2	R	The number of received dr signal for CH3 in M2
M25_cnt_in_L1M	0xA3	R	The number of received dr signal for CH4 in M2
M26_cnt_in_L1M	0xA4	R	The number of received dr signal for CH5 in M2
M27_cnt_in_L1M	0xA5	R	The number of received dr signal for CH6 in M2
M28_cnt_in_L1M	0xA6	R	The number of received dr signal for CH7in M2
M31_cnt_in_L1M	0xA7	R	The number of received dr signal for CH0 in M3
M32_cnt_in_L1M	0xA8	R	The number of received dr signal for CH1 in M3
M33_cnt_in_L1M	0xA9	R	The number of received dr signal for CH2 in M3
M34_cnt_in_L1M	0xAA	R	The number of received dr signal for CH3 in M3
M35_cnt_in_L1M	0xAB	R	The number of received dr signal for CH4 in M3
M36_cnt_in_L1M	0xAC	R	The number of received dr signal for CH5 in M3
M37_cnt_in_L1M	0xAD	R	The number of received dr signal for CH6 in M3
M38_cnt_in_L1M	0xAE	R	The number of received dr signal for CH7 in M3
Read_out_mask_L	0xAF	W/R	The low 16 bits for trigger mask.
Read_out_mask_H	0xB0	W/R	The high 16 bits for trigger mask.
L0_confirmed_cnt	0xB1	R	The counter for L0_confirmed
L0_inhibition_cnt	0xB2	R	The counter for L0_inhibition

Some tricks:

At the moment, the TRUs in M2 are ready for PHOS trigger test. So write 0x00 to register 0x1b: rcu-sh w 0x1b 0x00

Also the read_out_mask should be set, rcu-sh w 0xaf 0x8340

Go to /mnt/dcbrw/

Type rcu-sh b read_counter_M3_in.scr, you can get the trigger number in 10 seconds for each TRU:

Address	TRUs	the corresponding bits for mask on (the mask register is 0x1c)
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

After you set the threshold to 0x60, you have to check the these counters by type rcu-sh b read_counter_M3_in.scr

If you get answer like this:

0x38 : 0x3ee
0x39 : 0x1
0x3a : 0x2
0x3b : 0x789
0x3c : 0x2
0x3d :0x1
0x3e :0x3
0x3f : 0x689

Please choose TRUs which have small number for test.

In this case, you have to write 0x76 to address 0x1c: rcu-sh w 0x1c 0x76

If there are only one or two TRUs have small number, please reset the electronics and try again.