CTU firmware for integration test in Padova

As CTU_v0 has some design issue with the mini-WR interface, the VIO is used to control the CTU_v0 in this firmware. The control interface through IPbus is also provided using the link0 of the first SNAP12 tx/rx pair.

Registers and their function are show in table 1. See below for detailed specification.

Name	Width	Function
ch_mask	14	mask the channels that you don't want to take in to account when calculate nhit
threshold	16	trigger threshold for physics trigger
trig_mask	7	trigger source mask
force_trig	1	manual trigger input (ignore dead time setting)
ch_sel	5	select channel to be shown in the ILA
period	32	period setting for the periodic trigger
sma_sel	1	select source of SMA[1]
window	5	trigger window

- ch_mask: '1' to mask the corresponding channel. During the test, most of the channels will be empty. To avoid un-wanted noise from un-connected/noisy channels, set '1' to corresponding bit. ch_mask is actually 168 bits wide, cover all 168 BEC channels. As we only plan to use 2 BEC, the high 154 bits are constant '1' in the firmware.
- threshold: physics trigger is generated by comparing the nhit and threshold. If the calculated nhit is larger (>) than the threshold, a trigger will be generated. This register has default value 0 and should be set after power up.
- trig_mask: 7 trigger sources are provided by the firmware. If 1 or more sources are not needed, set its trig_mask to '0'. For external trigger source, DIO channel 3 and 4 are recommended.
  - trig_mask[6]: external trigger from DIO channel 4
  - trig_mask[5]: external trigger from DIO channel 3
  - trig_mask[4]: manual trigger from VIO
  - trig_mask[3]: external trigger from SMA[4]. Be careful when use this, should use LVCMOS18 standard!
  - trig_mask[2]: external trigger from SMA[3]. Be careful when use this, should use LVCMOS18 standard!
  - trig_mask[1]: periodic trigger mask
  - trig_mask[0]: physics trigger mask
- force_trig: rising edge of this register will generate a manual trigger.
- ch_sel: select RMU channel to be shown in the ILA. Range from 0 to 23.
- period: period setting for the periodic trigger. The default value is x" 3B9ACA0", generates 1Hz trigger.
- sma_sel: source for SMA[1], when '0', the SMA[1] output the original PPS from mini-WR, otherwise output the re-generated PPS.
- window: set the trigger window for physics trigger. The default value is 0, range from 0 to 31.

Trigger sources provided are as follow (sequencing in MSB to LSB of trig_mask):

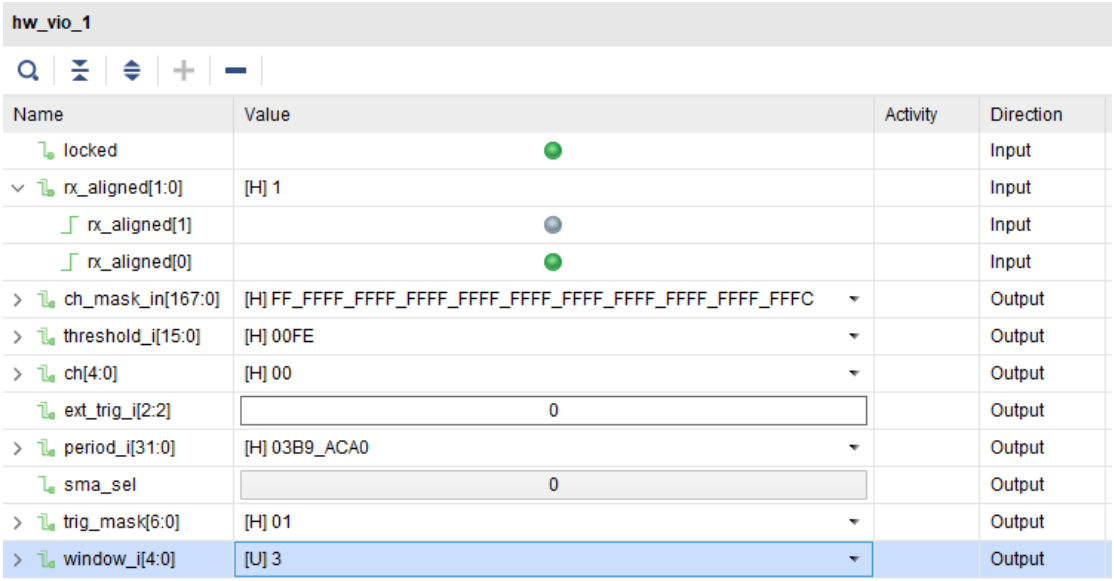
- force trigger: will cause a trigger certainly. When set this trigger during dead time, the FSM will wait until the dead time passed and give the trigger.
- external trigger: trigger from channel 3 of the FMC_DIO (could accept voltage range 2.5V to 5V TTL or LVCMOS signal).
- manual trigger: can be set through IPbus interface, used as the enable signal of the GCU test pulse.
- periodic trigger: period is set by the "period" register, which is the value * 16ns
- physics trigger: hit sum trigger.

To enable a trigger source, set the corresponding bit of trig_mask to '1'.

When there's a trigger, the CTU will send the trigger together with its trigger type to the BEC. If the BEC find the trigger type is manual trigger, it will send a "auto_trigger" broadcast command to the GCUs. Otherwise, the BEC sends the trigger type and the last 40 bits of the trigger timestamp to the GCUs.

An example setting when using VIO is show below (set "use_vio" to '1'):

- "locked" shows the clock generator status;
- "rx_aligned" shows the status of the link with RMU. Here only channel 0 is connected so rx_aligned[0] is on;
- "ch_mask_in" masked all channels except the first two;
- "threshold_i" sets as x"FE", hit sum larger than this value will cause a physics trigger;
- "period_i" is the default value which is 1Hz;
- "trig_mask" is 1, only physics (nhit) trigger is enabled;
- "window_i" is 3, the trigger logic will add the hit number in the last 3 clock cycles to make trigger decision, to compensate the TOF and link un-even.



To use the IPbus interface, connect the link0 of the first SNAP12 tx/rx pair to an optical switch then connect a computer to the same switch. The IP address is written in the firmware, currently is 192.168.10.32.

Open a terminal and run the ctu_console.py scripts. Frequently used functions are

implemented in the menu.