CMS Tracker APV test beam trigger card circuit description

General remarks

General NIM (Lemo connectors)

For the NIM as to TTL the MC10125 is used. Although this is not the best receiver we don't expect long cables fort these inputs so it should be sufficient. The advantage is that there is not an extra receiver necessary.
In principle the MC10125 is a differential ECL level receiver but by applying an other reference voltage on one of the inputs it can be used as a NIM level receiver. This is done with a voltage divider between the normal ECL reference voltage and ground. Take care that NIM signals are defined active when the voltage is negatie (because it's in principle a current logic).

! = not in logic function
+ = or
. = and
ppx = pulse positive of x ps

Values:
xE means x ohm
xK means x Kohm

Termination 50 ohm to -2V

implemented as divider between 0 and -5V
Gnd --- R1 ---x--- R2 --- Vee
R1/(R1+R2) * -5 = -2 (-2.1)
R1*R2/(R1+R2) = 50E   (51 E)
R1= 91 ohm (E12), R2 =120 ohm.   Ivee =24 mA , Pr1 =45mW   Pr2=70mW

The circuit drawings:
In this directory you can find the circuits drawings. Some circuits are distributed over different sheets, sorry for this.

Clk_trig.sch

This circuit takes care of the clock choise and the trigger clock synchronization and the aceptance of the trigger within a certain time window.
Except the E_clk_in all the inputs for this circuit come from the Lemo connectors at the inputs and converted to ECL level (see input option for clk and orbit input).

coincedence logic

The input of SC1 is connected to the positive input of the differential input of U22 (pin12). Because NIM logic is a negative level logic Cout (pin 18) = ! SC1 . Same is true for SC2, Dout= ! SC2.
(If T_SC2_ENABLE is active see below) !SC1 and !SC2 signals are (wired) ored (after conversion) .
So Cout=Dout= !SC1 + !SC2
The and gate U33a inverted the signal so the output will be !(!SC1 + ! SC2) = SC1 . SC2.
To be independed from the pulse width of the input pulses, the inverted output of U33A (so (SC1 . SC2) is delayed and inverted with ANDgate U33B. This signal is anded with the original signal (SC1 and SC2) in gate U33c. This result in a pulse with a width of on gate delay (~400 ps) at the output of U33c. This signal is used in the trigger synchronization circuit.

The SC2 input can be used as an input for the second scintillator or can be disabled by de-activating the T_SC2_ENABLE input (TTL from the logic control chip) . See sc2 input options.
If T_SC2_ENABLE is active the not SC1 and not SC2 signals are (wired) ored (after conversion) . If T_SC2_ENABLE is inactive the logic level not SC1 will appear on the output of U22 (pint 18 ,19). In this case an coincidence logic (if needed) has to be realized outside the trigger card (by example with NIM modules). This later setup is assumed to be the standard setup for a test beam. This gives more flexibility to build a good coincidence logic. The possibility to use SC2 is only for less critical application or in case scintillator already provide nice NIM pulses with a well defined pulse width. .

Below the trues table if T_SC2_ENABLE is active. In case T_SC2_ENABLE is inactive the yellow lines in the table will not be possible

SC1	SC2	Cout	!Q0	Q2
0	0	1	0	
0	1	1	0	
1	0	1	0	
1	1	0	1	pp400
pin 13 U22	pin 15 U22	pin 18 19 U22	pin 8 U 33A	pin 12 U 33C

sc2 input options

If T_SC2_ENABLE is low (inactive) I_{R11} = 4.7 / 910 = 5.1 mA
U_{R13} = U_{R12} = 560 * 5.1 = 2.8V ==> I_{R12} = 15.8mA. So SC2 is '1' (NIM logic).
So Din < !Din so "Dout" (if it was not conneted to Cout) = 0.

trigger synchronization circuit

This circuit has two purposes:
- align the trigger (coincidence of the scintillator) with the 40 MHz clock

- accept only triggers (rising edge from U33c) within a certain time window in respect to the clock. The circuit is based on the circuit that is used in the present tirgger card of the CMS tracker test beam. On a rising edge on the pulse comming from U33C (conincedence logic) connected to D input of flip flop U 30 (pin 3), the Q(pin7) goes low and the !Q(pin 6) goes high. This situation can only be changed by a set pulse (on pin 1).
Normally !Q output is connected to the set input of the device via an lemo cable of xns between the time_window_out and the time_window_in lemo connectors. So xns after the !Q input became active it goes low because the set input pulse.

If rising edge of the clock pulse comes within these x ns, the !Q_{U31} goes high and in the next clock period Q_{U32} goes high. This signal goes (after ECL-> TTL conversion) to the trigger logic. If the rising edge comes not within the x ns Q_{U30} goes low and if then the clock arives !Q_{U31} goes (or stays low) and the same is true for Q_{U32} after the next clock pulse so the trigger logic will not see the trigger.

To have a minimum delay there is a jumper between pin 6 and 1. Most properly this will not work. But perhaps with a RC we can achieve a shorter delay than with the lemo cable. The lemo cable is terminated with R5 and R61 (50 ohm to -2).

It is not garanteed that the hold time for U31 is resepected. This is alway a problem of a-syncrhone signals. This can result of meta stable situations of the device U31. The standard solution is a second flipflop (U32).

To have the possibility to work without a window (this means without lemo cable) U30 has to be reset after a trigger is accepted. This could be done to generate a pulse from Q_{U31}. The problem is that Q_{U31} should go low before a next pulse can be generated. With the limitations of the gates this was not possible to guarantee when two trigger pulses came within a clock cycles. Therefore it was decided to set U30 every clock period with a pulse generated by U33D and U33E. The consequences is that there is a window of ~600 ps , ~300 ps after the rising edge of the clock where triggers are not seen. So a real 2 ns acceptance range is not possible.

The orbit input

The Orbit input can have different input standard depending on the mounted components (see below). U22 is a receiver IC.
Pin 9 of U22 is the reference pin, The voltage is set to Vbb if a ECL input signal is used , to -.04 V if an NIM input is required. Bout (pin 4 , U22) gives the logic ECL level of the oribit signal in case of a NIM input signal and the inverted signal in case of a ECL input signal. This signal is converted to a TTL signal by U16. This signal goes to the trigger logic.

Input options for orbit input

For the clk and orbit input the following options are possible:
- NIM input orbit
  R44 none R45 50 ohm R42 680 ohm R43 270 ohm C64 0 **ohm**
- ECL input orbit
  R44 =82 ohm R45 130 ohm R42 0 ohm R43 none C64 0 **ohm**
- ECL -AC input orbit

R44 =82 ohm R45 130 ohm   R42 0 ohm R43 none C64 .1uF

Clock circuit

The clock used for the tigger logic and the trigger synchronization circuit can be the internal clock oscilator or an external clock connetected to lemo connector clk in (clk2). The external clock is converted to a ECL signal using U22(connected to pin 5 = !Ainput). Also the internal clock oscilator signal is converted to an ECL signal. Both are wired or together in this circuit.
If the external clock is connected the clock selection logic in the VME interface should take care that the E_clk_int signal is low. This is indicated by a switch on the board. If the internal clock is used no external clock should be connected to the input.
During the layout it became clear that the clock trace become to long and to much branches. Therefor U40 is introduced.
U16 , channel A is used to convert the clock to a TTL signal what will be used to control the trigger logic.

Input options for clk input

We tried to combine three input standards selctable with replacing resistors and capacitors.
The problem is that if no clock is provided the output at pin 3 of U22 should be low. Therefore we needed an extra signal to force the output low.
(Vbb = -1.3 V)
NIM input:
R60 = 1K R58 = none , C98 = 0 OHM !!! C63=0 OHM!!
R18 = 270 E , R20 = 3K ==> Ain = - 5 * (270/ (270+3K)) = -.41 V
R41= 51 E R19 =none
If Clk2 = '0' (0V) !Ain (pin 6) > Ain (pin7) ==> Aout (pin 3) = '0'
So if noting is connected A_out = '0' so the signal from the E_clk_int is

ECL input
R41= 120 E , R19 = 91E (standard ECL bias) C63=0 OHM!!
R20= 3K3   R18=3K3 R60 = 100E R58 =2 K , C98 = 10nF
If Clk_enbl = low (Q2 is "open"), Ain = -5 + 10 * (3.3 /(2+ 3.3+3.3) = - 1.2 V
So normal ECL ref
If Clk2 is not connected !Ain = -2 V ,
If Clk_enbl = high (Q2 is "closed") Ain = -5 + 5.1 * (3.3 /(3.3+ 3.3) = -2.45 V
(Internal in the device the transistor connected to Ain is active so the internal current should give no problem).
Ain < !Ain ==> A_out = '0'

AC- ECL input
R41= 68 E , R19 = 200 E => !Ain   = -1.26 V (so around ref what is o.k for the clock).
R20= 3K3   R18=3K3 R60 = 1K R58 =2 K , C98 = 10nF C63= 10nF
If Clk_enbl = low , Ain = -5 + 10 * (3.3 /(2+ 3.3+3.3) = - 1.2 V
So normal ECL ref
If Clk2 is not connected !Ain = -1.3V ,
If Clk_enbl = high (Q2 is "closed")   Ain = -5 + 5.1 * (3.3 /(3.3+ 3.3) = -2.45 V

Ain < !Ain ==> A_out = '0'

VME_DBUFF

This a simple bi-directional buffer. The B side (pins 11 .. 18) are connected to the VME bus. The other side is connected to the VME decoder and the trigger logic CPLD's (local bus).
If D_ENBL is high all outputs are high impedance. Therefore R33 takes care the VME bus (and local bus) are high impedance if the VME decoder output D_ENBL is high impedance.
If D_DIR is low the B bus (VME) is the input and the A bus (local) is the output .

The trigger logic FPGA

The trigger filter "fifo"

The name is not correct anymore. It is now a state diagram: The trigger filter state diagram.

The aim of this filter is to select the trigger depending on the trigger history.
The selected trigger depends on the setting of the mask. This mask is set by writing in the det_reg_set register.
Starting from the idle state.
As long no trigger arrives the machine stays in it's idle loop. After receiving a trigger it goes to W1.
If t = 0 (no second trigger) or none of the mask bits is set the machine goes to w2
so if 10 or 1x (mask = 00)
if t=0 (no third triiger) or mask = 00 the machine goes to state t1 what generates the triiger
so if 100 or 1xx if maks = 00 a trigger is generated .

We can follow al the possible paths through the diagram.
We get this

```
mask = 00     1xx     =>    00100.


maks = 01     100     =>    00100
              11x     =>    000100
              101     =>    0000100


mask = 10     100     =>    00100
              110     =>    000100
              111     =>    0000100
              1111111 =>    00001001   w1->w3->d2->d1->T1->d2->d1->T1
mask =11      10x     =>    0000
              11xxx    see maks 10
```

so:
mask 00 : always generate a L1 on the first arrival of a trigger
mask 01: always generate a L1 but prefer the second trigger if there is one or the third trigger if there is one and no second trigger
mask 10 : always generate a trigger and prefer the third trigger
mask 11 : if there are no two successive triggers don’t generate a trigger otherwise follow 10.

reset circuit

For the reset of the CPLD’s a MAX is used.
We only use the capability to activate the reset if the power supply comes under a certain voltage. The 3.3 and 2V are not monitored.

The calculations for the levels:

VL: Lower threshold (at "power down" when the reset signals goes low , active)
VL=4.5V
R1=R2(VL/1.29 - 1)
R2=200
R1=200(4.5/1.29 - 1)=498
After measurement R1->485 VL->4.5V

VU:Upper threshold (at " power up" when the reset signals goes high, inactive)
VU=4.75V
R3=(1.29*R1)/(VU-VL)
R3=(1.29*498)/(4.75-4.5)=2570
After measurement R3->2.4K VU->4.75

trig logic

bus control block

Several internal busses and external busses are connected to this block:

data busses:

- d16 16 bits bidirectional VME data bus (after data and direction buffer for the moment this is the communication channel.
- h_data 16 bits bidirectional data bus connected to the vme logic cpld. For the moment not used.
- tine_stamp 16 bits internal bus, input for this bus. Future development will generate a time stamp on this bus
- ext_data 7 bits output to write data to the different registers and the external eprom
- int_data 7 bits input data from different counters , registers and the externa eprom. This is a very

slow bus because all connected ports have to be multiplexed because Altera doesn’t has intenal busses.

address busses

- h_add 12 bits address bus , input. The twelf last significant bits of the latced VME address.
- v_reg_add 12 bit address bit output to address the internal registers and the memory location of the external eprom.

Practical notes for programming the devices

To program the device under dos control:
use the program jam

for programming the max 7192

jam -v -aprogram path\vme_interface.jam (parallel port is assumed)
jam -v -averify path\vme_interface.jam

to create the jam file

choose device emp7192s with file path\vme_interface.pof
choose device epf100ke with file none
choose device epc2 with file none

so in the window shoud be:

1 EPC2	none
2 EPFF10K100E	none
3 EPM7192S	path\path\vme_interface.pof

choose the output file name (vme_interface)
file format .jam (DEDEC STAPL Format).
then ok to generate the file.

for programming the flex 10 k directly

jam -v -aconfigure path\trigger_sync_fpga.jam
After switching of the power the fimware is lost (the firmware from the epc2 is loaded).
verifing is not possible.

to generate the jam file
choose device emp7192s with file none
choose device epf100ke with file trigger_sync_fpga.**sof**
choose device epc2 with file none

choose the output file name (trigger_sync_fpga)
file format .jam (DEDEC STAPL Format).
then ok to generate the file.

to program the epc2 (for statup firmware for the 100K)

jam -v -aprogram path\trigger_sync_fpga_epc2.jam
am -v -averify path\trigger_sync_fpga_epc2.jam

to generate the jam file
choose device emp7192s with file none
choose device epf100ke with file none
choose device epc2 with file trigger_sync_fpga.**pof**

choose the output file name (trigger_sync_fpga_epc2)
file format .jam (DEDEC STAPL Format).
then ok to generate the file.