Guide to DAQ operations: (supposing cold start == crate reboot || pcs reboot && etc)*

1. You will need to run PVSS on the windows PC (pcveloassembly3, on the left). Start -> All Programs -> PVSS II 3.0 -> PVSS II Project Administration . This starts PVSS and opens a window "PVSS Project Administration...".
2. Select VeloECSprodCBCCpcNew and click Start Project (green light).
3. you will need the Specs to be running. There are some scripts and there will be even more of them located inside ~lhcbvelo/AssemblyLabDaqScripts/. execute./RunSpecsv2r0 from there. make sure that the firewall is stopped on the specs server side (type: sudo service iptables stop).
4. on Vision_1:SingleCB panel, make sure you selected the right SPECSnode (named pcveloassembly4) and the first one of the SPECS ports Master ID, which is the one we usually use (05 pci:1 port:1, in the assembly lab). CB address should be 2 and SPECS speed 128 kHz.
5. (this step is not needed but we leave it just temporarily here) You have to start the CCPCs. Sit at pcveloassembly4 as lhcbvelo and do ssh tfc@odinv220. Then, in odin, execute the script: source .VeloSetup and leave that window aside (do not close it). You need to wait till the automatic loading of FPGAs is done.
6. It's a good time to start the TELL1 daq program. just type ./startAssemblyLabTell1s_digitisationScan from the AssemblyLabDaqScripts directory.
7. Start a couple of windows to monitor the tell1s. ssh velo@tell029 and ssh velo@tell030. Then, in each tell1 window type ./startConsole.
8. If everything is fine you should have on pcveloassembly3 your PVSS panel open called SingleCB. From the Temperatures tab, choose the ELMB and press load table. Then press Start. This should start temperatures read-out and logging.
9. Search for the PVSS TFC Vision_1:fwFSM panel (FSMOperate.pnl). Click on +TFC to get the list of ODINs. Pick Partition_OdinV2_20 and highlight it. Right click, wait, and choose View. If the panel is not "taken" i.e. the little lock in the box "State" next to Partition_OdinV2_20 is open and not green, then click on it and do Take. It should go green. Then you can operate.
10. If you are in the situation where the little lock is red, and you are told that you are in local, then you have a problem from some previously hanging session. Close the Partition_OdinV2_20 window, go back to the PVSS panel, right click on the OdinV2 _20 and click start/restart tree. Then view the window again.
11. In the ODIN Initialisation panel click on "Select Recipe" and from the Cache choose Kazu's favourite: TEST/ODIN/R_All/VeloTestPulse1L0ConsecDigitisationScan. Click on "set recipe". You should get a yellow NOT_READY. On the top panel ("state") click configure and choose send.
12. On the right side of the panel, in the boxes L0 trigger, Command and Other tick all boxes which have a green dot, untick all the others. Configure again.
13. Now you can start the run.
14. The Specs server bar should be green (if not just execute RunSpecsv2r0 again, wont harm). You should be ready to use the Master control Panel.
15. from the Master Control press the button off to cascade down to GetReady. this will configure the whole control board and check which cables are connected to something.
16. press the LV on. this wil enable the low voltages from the repeater boards tu the hybrid.
17. press on the off states to go to on. this will configure the beetles.
18. start the triggers from the Odin Control panel or the TFC partition (the one that pops up when you load the recipe)
19. go to velo@velotestNOSPAMPLEASE.daq.lhcb this is the DAQ pc (you need to be a velo expert to get inside). type sudo su- to become root
20. type source setupvelodaq. this will lead you to /home/velo/eventBuilder/writeEventsToBinary and load the paths.
21. optional step: to spy on on the ethernet card type tethereal -i eth1 and check if data of both tell1s arrive
22. to acquire data just type: ./takeData <filename string> <module name> <N events> it will creat the directory /scratch/lhcbvelo/assemblyData/<ModuleName> (if it doesnt exist), and copy the files to that place on pcveloassembly4. the <filename string> will have .bin added to it.

23. Analyse the data:
    1. (Open a fresh terminal) and do
        a. [pcveloassembly4] ~ > source assemblyVetraSetup from the AssemblyLabDaqScripts directory
        b. [pcveloassembly4] ~/cmtuser/Velo/Vetra/v2r3/cmt > vi ../options/fullsnapshot.opts
    2. change the two lines:
        a. "DATA='file:///scratch/lhcbvelo/assemblyData/Mxx/modxx_snapshot_ntp_whatever.bin' SVC='LHCb::MDFSelector'"
        b. OutputFile = "/scratch/lhcbvelo/assemblyData/Mxx/modxx_snapshot_ntp_whatever.root";
    3. Then, do
        a. [pcveloassembly4] ~/cmtuser/Velo/Vetra/v2r3/cmt > fullSnapshot
        b. [pcveloassembly4] ~/cmtuser/Velo/Vetra/v2r3/cmt > root
        c. root [0] .L ../macros/badChannelX.C
        d. root [1] badChannel("assemblyData/modxx_snapshot_ntp_invac1.root",29,'r',xx);
24. Put in elog the 4 noise jpg plots and the bad channel lists

If bugs or mistakes in this guide tell me kazu.akiba@gmailNOSPAMPLEASE.com and i'll fix it!!

PVSS

PVSS is the framework where we create programs to control the hardware. The most visible part of the PVSS are the panels, but other tasks running in the backgrounf are important too.

- You need to have PVSS running and with the right project. If the PVSS is not running then click **Start** -> **All Programs** -> **PVSS II 3.0** -> **PVSS II Project Administration**. Choose the **VeloECSprodCB** project, right-click and choose "Start project". It will take a minute or two before all the tasks are running thus be patient and watch the log file which opens as a very tiny window (make it bigger !) - proceed only after the log file ceases to report the tasks starting up and initializing.

CCPC (*Credit Card PC*)

The Event Supervisor board and the TELL1 board are controlled by Credit Card PC's: tiny, diskless PC's running Linux. You talk to them by logging in like to an ordinary PC running Linux thus by **ssh velo@pctell25** (TELL1 board) or ssh **tfc@odinv220** (Event Supervisor). To see if they are alive you can ping them: **ping pctell25**. When you power up a board its Credit Card PC boots via ethernet and this process takes a minute or two.

- A CCPC needs ethernet to boot, to get access to its file, etc. Without an ethernet connection and a proper configuration of the DHCP and other servers it won't work. If there is a problem with booting, contact Niko Neufeld.

- Credit Card PC's are not radiation resistant and thus they have to operate outside of the radiation wall of the LHCb. For more harsh enviroment a SPECS mezzanine is the right choice.

- PVSS can not talk directly to the CCPC: instead, the CCPC runs a DIM server with board-specific variables/commands and the PVSS talks to it via its DIM client.

- For the TELL1 board the CCPC DIM server is not yet fully developed and thus we need to talk directly to the CCPC by mean of (Linux) commands, scripts and programs.

SPECS

The Vertex Control Board is controlled via the SPECS bus and the data and commands are received by a

SPECS mezzanine card, which can withstand certain amount of radiation (it is based on a radiation-hard FPGA).

The PVSS can not directly use the SPECS bus: instead it uses DIM and talks to a **SPECS server** that runs on a (Linux) machine with a physical SPECS interface (a PCI card).

- With the PVSS project a big control panel **SingleCB** will start: there (upper-left corner) you need to specify the SPECS server machine: **pcveloassembly4**, the interface **05** (first on the list) and the slave address of the SPECS mezzanine on the Control Board: **2** (it is set with the jumpers on the SPECS mezzanine). When done you will be ready to control the Control Board, the Repeater Board(s) and the Beetle chips.

- You may need to (re)start the SPECS server as it crashes often when something goes wrong with the SPECS hardware (like you forget to provide the clock).

You can check which version of the SPECS server is running by typing **sudo su -** followed by **ps -aux | grep -i specs**. If it is the wrong version then kill the specs server and the dns server with a command like **kill -9 id** Do the same for the dns server, type **ps -aux | grep -i dns.** In such case login to **pcveloassembly4** as **lhcbvelo**, type **cd /home/NewerSpecsServer/fwSpecs1.6/bin** and then **sudo service iptables stop** (kills the firewall) followed by **./RunSpecsv2r0** - this will open two xterm windows with the logs of the DIM DNS and SPECS servers.

- The Linux version of the SPECS server is very fragile: if anything goes wrong with the hardware or a command is not successfull it simply dies and so you have to restart it. Windows version once started runs forever.

- In case you can not get the SPECS communication right, you need to reset the SPECS mezzanine or power cycle the whole crate.

TELL1 board

TELL1 boards receive, digitize and analyze in real time the data sent over from the Beetle chips on the hybrids over a 60m analog link.

- Log into the CCPC (*Credit Card PC*) that controls the TELL1 board: **ssh velo@tell029** (25 is the number found on the front pannel of a TELL1 board). Use the appropriate expert's password.

- execute: **source OneMHzSetup** - this will lead you to the correct directory and run the setpaths script as well.

- read the config file. Type **ls -ltr** to see the most recent .cfg files and there should be one specified to be used in the lab15. important settings are the ECS Trigger type, which should allow you to specify zero suppressed or non zero suppressed data, and GBE ports select (check if selected the correct setting that should lead to the Gigabit port connected to the DAQ PC or the switch.)

- a hint for running emacs try **emacs -nw**

- start the ./ccpc_daq program. give the config file as the argument: **./ccpc_daq tell125Lab15ZSNPFGen.cfg** This should initialize the TELL1. you should press enter a couple of times and the TELL1 will be waiting for triggers. If things do not work, like you see error messages about no I2C acknoledge from the FEM, reset the TELL1 board. If you do not know how to reset then cycle the crate power (in this case reinitialize the other hardware too).

Readout supervisor (Odin)

Readout supervisor named Odin (after a Swedish god ?) controls the detector(s) readout by sending the clock, reset, trigger and test pulse signals on an optical TTC (*Timing, Trigger and Control*) link. The signal is received by TTCrx chips located one on the TELL1 board and the other on the Control Board. The Control Board then propages the signals to the Beetle chips via the Repeater Boards.

The Odin's TTC signal passes through a **NIM -> TTC converter** (in a "portable" VME crate above the Readout Supervisor) thus make sure the crate and the converter are turned on.

- Log in to the Odin CCPC with **ssh tfc@odinv220**. The name of the Odin is visible on the label on its CCPC in the lower left corner of the board. Again you will need the expert's password.
- source .VeloSetup* - it displays the message "Starting up server /TFC/OdinV2_02/SERVER". Leave it in this state.
- Start the PVSS panel that controls the Odin (for running PVSS see the top of this page): go to the **Project Console** and find the manager: **PVSS00NV -data lbontfc01.cern.ch**, right-click and choose **Manager start**. Note: that panel sometimes starts hidden behind othe panels.
- On the left select **None**, a recipe "Stored in Cache" (like **R_All_VeloTestPulse1L0Consecutive**) and click **Find Odin**.
- On the right find our **OdinV2_20** on the list and click it so that it becomes selected (you *must* click it !). Click **Configure System** and a run-control panel will pop up.
- Control the system state (**RUN_ERROR**, **RUN_READY**, **RUNNING**) - if you can not (no software is perfect) then control only the **OdinV2_20** (that is enough for our needs). If still no effect, exit the Supervisor panels (*always* use the **Exit** button !) and start them again. If you get desperate call Richard Jacobson...
- To play with parameters (like timing) click **ODIN not ready** in the **Configuration** frame and the panel with parameters will pop up. Choose various tabs for example **Triggers 2**. To make the parameters editable click **Current All** in the **Initialize** frame. Apply the new parameters with the **Apply** button.
- In the **Triggers 2** tab: to see the test pulse on the Beetle output you need to set the **Calib. trigger delay A** to 172. To send several consecutive triggers set the **Calib. trigger window A** to 1, 2, etc.
- Your goal is to make the Readout Supervisor send all the timing needed by the sensors (Clock, Front-End Reset, Trigger, Test Pulse). You can check those signals on the various connectors on the Control Board.

Control Board

The Control Board take the TTC signal from the optical link and distributes it to up to six Repeater Boards (and detector hybrids). It controls as well the power supply and talks to the Beetle chips on the I2C bus. Use the PVSS SingleCB panel to get access to these functions. For that panel to work the SPECS server must be running on a PC with the SPECS interface.

The SingleCB is made of several sub-panels representing different parts.

- **SPECS** sub-panel controls the SPECS mezzanine: **Initialize I/O** sets the I/O pins and their directions in the right state (normally to be done only after a power-up or reset). **Clock source** selects where the clock is taken from. **TFC enable** sends the clock to selected hybrids. **PwrOn Rst** sends a reset signal to all Beetle chips on a given hybrid. **Refresh** button reads and displays the state of the SPECS

mezzanine.

- **LV** sub-panel controls the low voltage power supply for the repeat boards and the hybrids. You can turn on all hybrids and all voltages or power only one hybrid with selected voltages. You can read back the voltages and LV regulators status to make sure that the power is OK.

- **Delay25** sub-panel control the timing signals: for every hybrid and every signal you can choose the delay in steps of 0.5 ns. You can as well enable or disable particular signals.

- **Beetle** sub-panel talks to the Beetles on the I2C bus: you can read registers of a particular Beetle chip, modify them and apply the changes. You can choose nominal values and apply them to all the Beetles on a selected hybrid.

- New: try **MasterControl** tab to setup the control board with a single click: the command "Get Ready" with load default values everywhere. "LV ON" will turn on the low voltage to the hybrids and then you can turn on or off Beetles on individual hybrids.

High Voltage (bias for the silicon)

High Voltage (HV) is suplied by a HV power supply from the **iseg** german company. The power supply is controlled via a CAN bus (gree cable) and a program supplied by the company which runs on Windows. In the test beam this program and the USB <-> CAN bus interface are installed on the **lbtbxp02** PC. The program is called **isegCANHVControl** and can be found on the desktop or via **All programs**.

to be continued ...

Interlock

The following steps can be performed to check that the interlock is working.

1. Both the LV power supplies should be on for a good test. But VERY IMPORTANTLY note that the interlock is actually powered from the Bellegarde power supply. Which should never be switched off

2. In the quiescent state you should see 0.03 A consumed on both supplies

3. Put the "too warm" signal, i.e. the one labelled 3.3K into any input on the temperature board

4. You should see the current drop to 0.01 A. You should also see on the PVSS display the too warm temperature displayed.

5. You can recover the interlock by plugging the "too cold" signal i.e. the one labelled as 51K into the bottom left position.

6. REMOVE it

6 - Run the panel, select None, look for recipe and select odinv202. press find odin and configure system. Then another panel for the odin operation will run. make sure that the settings are correct on the configuration->Odin not ready button. Look for the triggers 2 tab and look if the setting calib trigger delay is set to 172 as Pawel and I found out yesterday. (with the beetle latency set to 160, ie 4us)

- Try to configure the control board and friends using the panel which just popped up. Choose your specs server, interface and state address. In the section of the panel called **Specs Central Control** choose the SPECSNode as e.g. pcveloassembly2, the MasterID as 09 (PCI: 2 PORT1). Or ask Lars. The CB Address is set by jumpers on the SPECS mezzanine on the control board. In the integration

lab it is 2. Select the SPECS speed to be 128 kHz.

- Click on the **SPECS** tab and click on the button **Initialize I/O**. The button will turn green. Check that FRIEA is sending clocks. You should see two green LEDS indicating the status of the TTCrx chip next to the input of the optical link on the control board. These lights should be ON. Go to the button **Clock control** and choose TTCrx as the clock source. Now enable the clock to the hybrid you are running by clicking on the appropriate checkbutton in the **TFC Enable** list.

- Now go to the **Delay25** tab and push the **Default** button. It goes green. Now set the testpulse delay - currently 15ns. Now push the **all hybrids** button. It goes green. This configures the delays for the timing signals and enables them.

- Now go to the **LV** tab. Now you should be very careful that the cooling is on as you will be supplying power to the hybrid. Select the checkbutton for **All** hybrids and click on **Set**. Now you should see some current consumption on the LV supply. Click **Read** on the LV monitor panel and you will see all the currents. They should be all green indicating no overcurrents.

- Now click **Beetle** tab. Select your Hybrid, and Base (48 is R and 32 is Phi) click the **Default** button, which will go green and display the default parameters, then click **Apply to hybrid** if you want it on all the Beetles, or just **Apply** if you want individual Beetles.

7- Configure the Control board to enable all the signals. Use pawel's panels for that. it should be easy to understand. You need to enable the control signals, the clock, L0 reset, test pulse and l0trigger. If the Odin is in run state you can check if the triggers come. it's the pin number 3 (the 2st resistor on upper side of the little test connector, test pulse should be the third resistor counting from one!!!)

8- Alright! good to go! the I would advise to open another terminal and connect to the tell1 again. do source OneMhzSetup again and cd ../tell1lib. on this directory you should find the ./console_tell1. run it! this will open the console. Press "m" to monitor and "m" again to extract the raw data from the Mep bank. this should only work if you enabled nonzero suppressed data!!! check the ADC values!

9 - If the adc values make sense go to the DAQ pc. velo@velotest. become root **sudo su-**. go to/home/DAQ/writeEventsToBinary . do source setpaths. Note that this pc is set up to readout only one tell1 (check that on the ebuild.cfg file. the board id should be the same as on the tell1). if you are ready to go, just run ./writeEventsWithOdinBank .then you ger an error message telling you the correct syntax.

10 - fine! we are ready to run the DAQ and make a nice binary file. if nothing happens there are a few things you might need to check. check the following points in no particular order.

- first the firewall should be forever disabled (if the specs is running the firewall should already be disabled) This can be checked by simply typing the command: service iptables stop (as root, obviously).

- no events coming? try to execute tcpdump -i eth2 and see if there are packets coming from the tell1.

- still nothing? check the ethernet cables... check if the tell1 GBE port is the same in the config file and in the cabling. check if the leds are up, otherwise this GBE may have died or is unavailable.

- if nothing comes check the counters written on the tell1 ccpc_daq program. The tell1 should be happily writing in some MBA rate. If not maybe there is no triggers coming or the data got stuck.

- If there are short and long packets try to see if the mep factor on the tell1 config file is the same as in the ebuild.cfg. if it till with the problem, try to stop the tell1 and restart the ccpc_daq.

- if there are still problems. cry for help. curse my name and call the guido and the online team. (that's usually

what i do. including cursing at myself)

11 - if you made a binary file you should be able to run vetra on it. that will be yet another chapter.

-- Main.jalocha - 16 May 2006

This topic: LHCb > VeloHowtoDAQ
Topic revision: r32 - 2007-10-29 - StephenWotton