VEAGAS FPGA mode documentation

Authors:
Mark Wagner
Hong Chen
Jayanth Chennemangalam
Glenn Jones

September 16, 2013

The modes selected for VEGAS are l8_lbw1, l8_lbw8, l1_lbw_1, h1k, h16k. Below are a rough guide to the shared memory and setup parameters.

Compiltation of the FPGA designs was done with the 14.2 Xilinx tools, Matlab 2010b, and the ska-sa fork of the CASPER libraries.

Currently, in the repository (https://github.com/casper-astro/vegas_devel), the model files are under mdls/<fpgadesign>. For designs using black boxes, compilation instructions can be found in the READMEs, but the basic idea is that before running casper_xps on the primary design, it's required to open up the mdl file for the black box, open the System Generator block and click generate. This generates all the necessary files for the main design to compile using casper_xps.

The most up to date mdls used are listed below, along with their corresponding bof files. The scripts listed below have been used and tested in Berkeley and specific to that location, but do give and example of the process for configuring the FPGA designs. The general steps for running a design are:

1. HPC shared memory setup
2. Program the ROACH2 FPGA and configure it
3. Run the vegas data acquisition code.
4. Send a reset to the ROACH2 FPGA

HPC shared memory setup

The general nomenclature is:

```
setup_<mode>_<bandwidth>_<numchannels>.sh
```

There are a set of pre-configured scripts in $VEGAS_DIR that match the above nomenclature. Internally, they call the Python script $VEGAS_DIR/bin/vegas_setup_shmem_lbw.py, the usage of which is given below:

```
Usage: bin/vegas_setup_shmem_lbw.py [options]
    -h  --help                  Display this usage information
    -a  --adcclk <value>        ADC clock rate in MHz
    -s  --nsubband <value>      Number of sub-bands in data
    -c  --nchan <value>         Number of channels
    -g  --gpuint <value>        Integration time on GPU in ms
    -t  --totint <value>        Total integration time in ms
```

Example - configuring shared memory for l8_lbw8, with ~23 MHz bandwidth per subband, dumping spectra every 2.5 ms:

```
vegas_setup_shmem_lbw.py -a 1500 -s 8 -c 4096 -g 2.5 -t 2.5
```

Program and configure the FPGA

** The SSG status bit order has changed from older versions to:
bit 3. BLANK
bit 2. SR1
bit 1. SR0
bit 0. (LSB) CAL
(See below for screenshot). Please see the READMEs in the corresponding mdls directories for more information.

Shared memory common to all designs:

sg_sync - manually send a sync pulse: 0b010100, 0b010101, 0b010101
arm - 0,1,0 - allows the next sync pulse to implement a reset
dest_ip - selects the destination IP address
dest_port - select the destination port
adcsnap0,1 - snap blocks of adc values
reset - sends a reset to 10gbe in case of overflow
trig - manual trigger for snap blocks
https://casper.berkeley.edu/wiki/images/1/18/Ssgprogram.pdf

l8_lbw8

note: l8_lbw8 uses the same bof file as l8_lbw1

* SSG status bit order is 3. BLANK 2. SR1 1. SR0 0. CAL

model files - mdls/l8/l8_ver106.mdl, cichbcic_core.mdl, cichbcic_core.vhd, cichbcic_core_config.m
boffile - bofs/l8_ver106_2013_Aug_25_2033.bof
configuration script - newscripts/l8_lbw8/l8_lbw8_conf.py
reset script - newscripts/l8_lbw8/l8_lbw8_reset.py
HPC binary - src/vegas_hpc/bin/vegas_hpc_lbw

l8 FPGA design -

snap blocks -

filtersnap
adcsnap0 - snapshot of adc 0, requires manual trigger and valid
adcsnap1 - snapshot of adc 1, requires manual trigger and valid

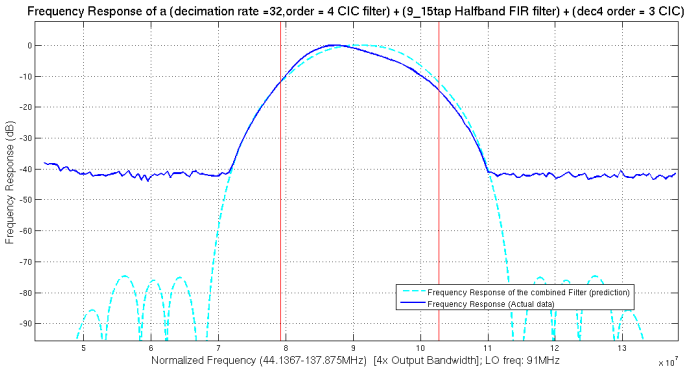
software registers -
mode_sel - 0 (select between l8_lbw8 and l8_lbw1 when the value is set to 1, the design will transmit all 8 subband)
s*N*_mixer_cnt (*N* = 0~7): set upper limit for the fraction of brams to be used. (see mixercic_funcs.py)

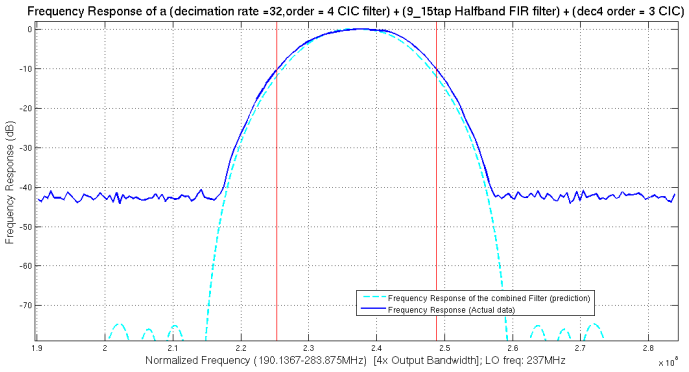
brams -
s*N*_lo_*m* (*N* = 0~7, *m* = 0~15): mixer brams (see mixercic_funcs.py)

note - mixerX_cnt on top level aren't being used any longer, so need to be removed.

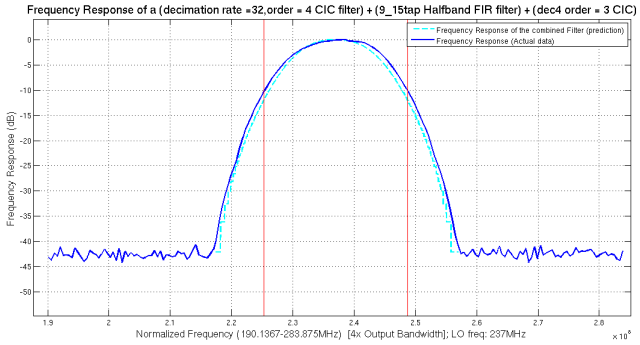
--Digital gain
For the l8 modes (l8_lbw1 and l8_lbw8) we have gain for each of the pols and the software registers are buried in the subband blocks. They should be labelled s[1-8]_quant_gain[1-2], where 1-8 is for each of the subbands, and 1-2 are the pols. After the gain, we slice off the top 8 bits of the 32 bit real and 32 bit imag values. The script we have in vegas_devel/scripts/l8_lbw1_lbw8/mixer_funcs.py sets the gains in the design and would be a good reference.

Frequency Response (also for l8_lbw1)





halfband coefficients quantized to FIX16_14 (as in the model files)
final response quantized to 8 bits (as in the hardware)



l8_lbw1

Clocked at 192MHz

model files - mdls/l8/l8_ver106.mdl, cichbcic_core.mdl, cichbcic_core.vhd, cichbcic_core_config.m
boffile - bofs/l8_ver106_2013_Aug_25_2033.bof
configuration script - newscripts/l8_lbw1/l8_lbw1_conf.py
reset script - newscripts/l8_lbw1/l8_lbw1_reset.py
HPC binary - src/vegas_hpc/bin/vegas_hpc_lbw

snap blocks -
filtersnap
adcsnap0 - snapshot of adc 0, requires manual trigger and valid
adcsnap1 - snapshot of adc 1, requires manual trigger and valid

software registers -
mode_sel - 1 (select between l8_lbw8 and l8_lbw1; when the value is set to 1, the design will transmit 1 subband)
s*N*_mixer_cnt (*N* = 0~7): set upper limit for the fraction of brams to be used. (see mixercic_funcs.py)

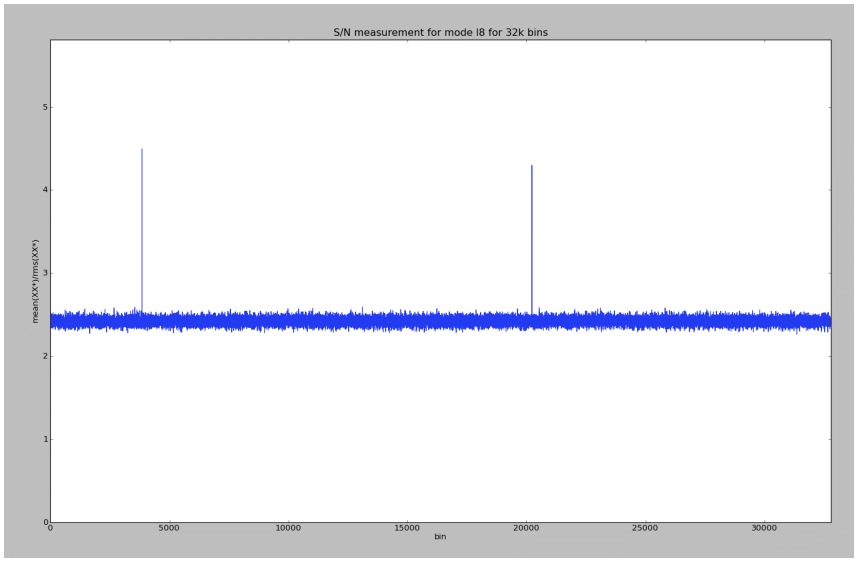
brams -

s*N*_lo_*m* (*N* = 0~7, *m* = 0~15): mixer brams (see mixercic_funcs.py)

note: the mixerX_cnt on top level aren't being used any longer, so still need to be removed.

SSG (Switching Signal Generator) - The SSG is the same for all modes and is documented here: https://casper.berkeley.edu/wiki/images/1/18/Ssgprogram.pdf

S/N (.5ms integration)



l1_lbw1

This design mixes by 4/16 of the adc clock freq, decimates by 8 on the FPGA, and performs the PFB in the GPU. Clocking the ADC at 1500MHz gives an effective bandwidth 656.25 to 843.75.

The current boffile targeted 192MHz.

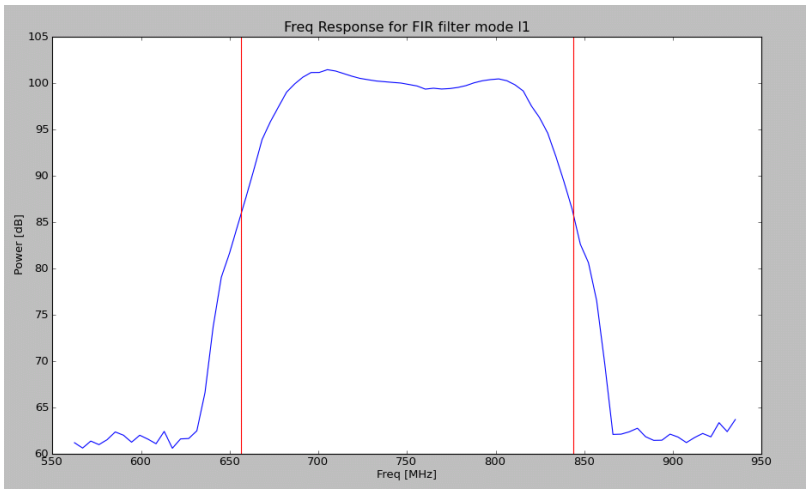
model file - mdls/l1/l1_ver106.mdl
boffile - bofs/l1_ver106_2013_Aug_24_1324.bof
configuration script - scripts/l1_lbw1/l1_lbw1_conf.py
reset script - scripts/l1_lbw1/l1_lbw1_reset.py
HPC binary - src/vegas_hpc/bin/vegas_hpc_lbw

snap blocks -
filtersnap0 - snap block after pol0 decimating filter and bit selection
filtersnap0_nogain - snap block after pol0 decimating filter, but before gain and bit selection
filtersnap1 - snap block after pol1 decimating filter and bit selection
filtersnap1_nogain - snap block after pol1 decimating filter, but before gain and bit selection
rshpsnap - snap block after the reshaper
packsnap - snap block after the packetizer

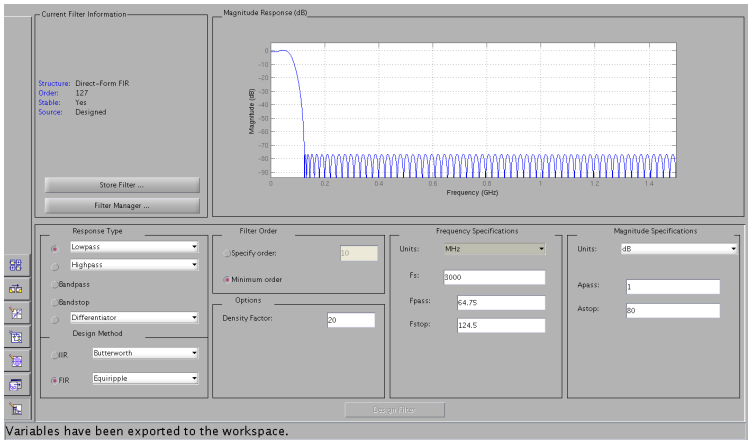
software registers -
gain - sets gain after the decimating filter
sg_sync - manually send a sync pulse: 0b010100, 0b010101, 0b010101
sg_period - this does not need to be set for mode l1
reset - only used to manually reset the 10 GbE in mode l1
trig - this is used to manually trigger all the snap blocks in the design
arm - 0,1,0 - allows the next sync pulse to implement a reset
gbe0_overflow - 1 if the 10GbE buffer is overflowing, 0 otherwise.
mixer_addr - this register is not used in the l1 more and can be ignored.
dest_ip - destination IP
dest_port - destination port

SSG (Switching Signal Generator) - The SSG is the same for all modes and is documented here: https://casper.berkeley.edu/wiki/images/1/18/Ssgprogram.pdf

Frequency Response for the decimating filter:



Theoretical frequency response using FDA tool:



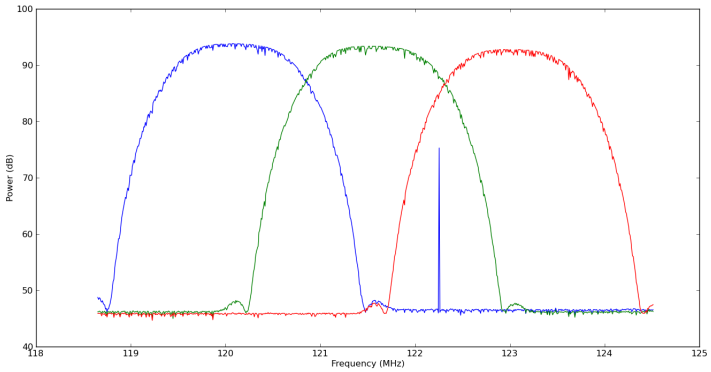
Digital Gain --

After digital filtering for FPGA design I1, we take the product of a software register 'gain' and the real and imag values, then select off the top 8 bits and output to a 16bit complex value. So for I1, the digital downconverter outputs a 64bit complex number. We separate the 32 bit real and 32 imaginary parts, multiply them by the gain, and output them as a 16bit complex value, with an 8bit real, and 8 bit imaginary parts.

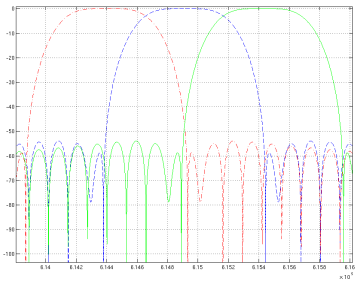
h1k

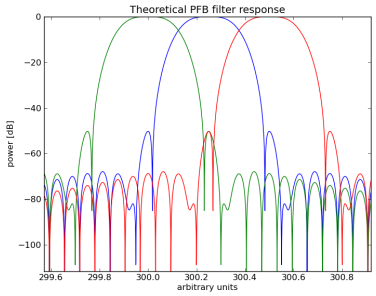
model files - mdls/h1k/h1k_ver103.mdl, fft11_bb.mdl, fft11_bb.vhd, fft11_bb_config.m,

pfb4t11_bb.mdl, pfb4t11_bb.vhd, pfb4t11_bb_config.m
boffile - bofs/h1k_ver103_2013_Aug_24_1500.bof
configuration script - scripts/h1k_hbw_conf.py
reset script - scripts/h1k_hbw_reset.py
HPC binary - src/vegas_hpc/bin/vegas_hpc_hbw
PFB bin response:



We decdied to use the -6dB response filter as it seems to have better adjacent band rejection and lower sidelobe levels. Below are the theorectical plots:

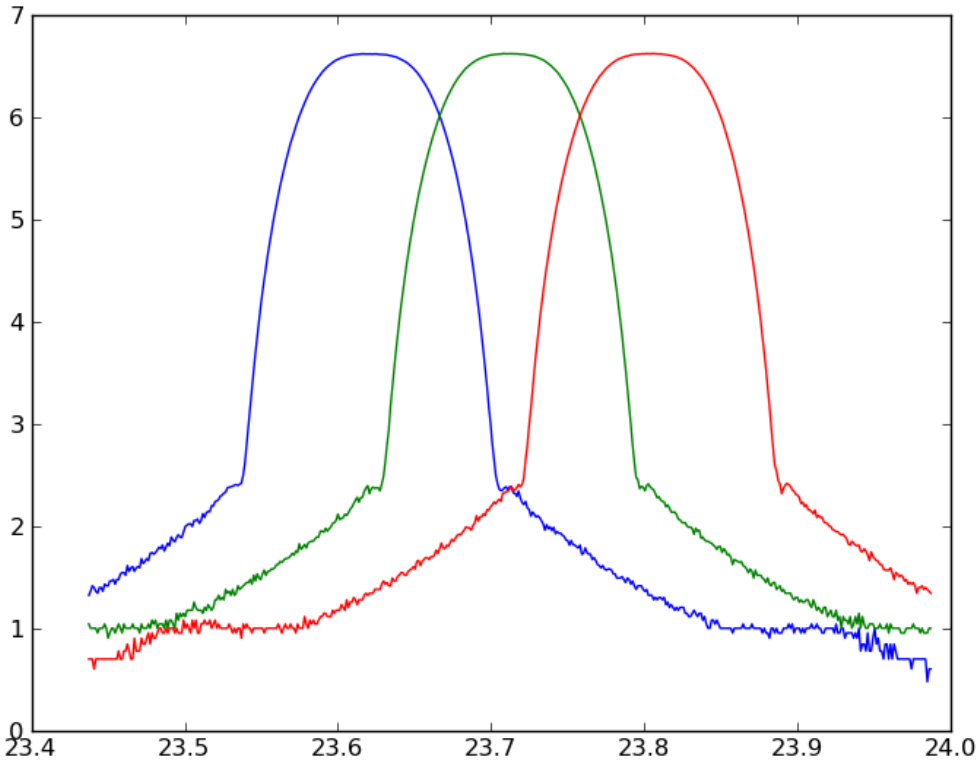




h16k

model files - mdls/h16k/h16k_ver103.mdl, fftwbr_core.mdl, fftwbr_core.vhd, fftwbr_core_config.m, pfbfirr_core.mdl, pfbfirr_core.vhd, pfbfirr_core_config.m
boffile - h16k_ver103_2013_Aug_24_1630.bof
configuration script - scripts/h16k_hbw/h16k_hbw_conf.py
reset script - scripts/h16k_hbw/h16k_hbw_reset.py
HPC binary - src/vegas_hpc/bin/vegas_hpc_hbw

PFB bin response:


7
6
5
4
3
2
1
0
23.4
23.5
23.6
23.7
23.8
23.9
24.0

SSG

Link to SSG specifications:
https://casper.berkeley.edu/wiki/images/1/18/Ssgprogram.pdf