VEGAS C-BANDxCHAN mode (BAND = 0800 MHz/ 1500 MHz and CHAN=0064 to 8192)

The latest firmware modules for VEGAS have been developed using Xilinx 14.7/MATLAB R2012b. The designs have been built for operations compatible with ROACH-2.

The VEGAS firmware building blocks, in the order of signal flow in ROACH-2, are as follows:

1) **ASIAA-ADCS**: Two interleaved ADCs, each clocked at 'BAND' MHz for the two polarizations P0 and P1, results in the input signal being digitally sampled at an effective rate of '2xBAND' MHz, satisfying the Nyquist sampling criterion. The resulting logic frequency of the Fabric is clocked at BAND/8 MHz. Each clock cycle yeilds 32 (16P0 + 16P1) samples, each 8 bits wide, in the offset-binary format. The signals are then converted to processible signed 2's-compliment format. In order to monitor the ADC outputs every clock cycle, the 16P0 and 16P1 8-bit streams are concatenated to form 128-bits P0 and P1 outputs from ADC0 and ADC1, respectively, and are logged using adcsnap0 and adcsnap1 blocks.

The ADCs also provide the framework with a 1 PPS, which is used along with a user-controlled ARM signal, to generate a Master-SYNC pulse, to synchronize the functioning of all the blocks in the design.

2) **Bus-expand**: The 128-bit P0 and P1 concatenated outputs from the ADC blocks are then split in order to re-obtain the 16P0 and 16P1 8-bit 2's compliment signals, to pass them on to further signal processing sub-systems.

3) **Polyphase Filter Bank**: The signal is then subjected to Fast-Fourier Transform (FFT) using the polyphase filterbank technique, instead of just a conventional Fourier transform operation, to alleviate leakage and scalloping losses. This is done by adding a PFB block, where initial signal conditioning takes place, before the FFT block, where the signal spectrum is generated for further processing/analysis stages. More information on this technique can be found in https://casper.berkeley.edu/wiki/The_Polyphase_Filter_Bank_Technique.

The 12-tap PFB block is set for 2*CHAN-points, and employs a Hamming window function, and a bin width scaling factor set to 0.95. The resulting output signal streams, each 18-bits wide now, are then fed to a 16-input 2*CHAN-point FFT blocks (FFT0 and FFT1), one each for the two polarizations. There are 8 outputs coming out of the FFT block, each 36-bit wide (18Real + 18Imaginary). It is to be noted that 8 channels of the total spectrum are being output, every clock cycle. This implies that it takes CHAN/8 clock cycles for one set of the entire spectrum to be generated. A single FFT_SYNC pulse (also output by the FFT block) marks the start of the first spectra being output. Subsequent spectra generation occur continuously thereafter.

4) **P0_SCALE/P1_SCALE:** The 8 outputs from the FFT blocks, for both polarizations, are first separated to 8 real and 8 imaginary components, and then scaled by a user-defined factor ‘SCALE_P0/SCALE_P1’. The scaled values are then shrunk to 8-bit real and imaginary parts, which are then combined together to yield 8 complex bitstreams, each 16-bits wide.

5) **MERGE_POL**: The outputs P0_S_i and P1_S_i (i = 0 to 7), from the P0_SCALE and P1_SCALE blocks respectively, are simultaneously fed into the MERGE_POL, where the polarizations are combined, using two levels of concatenations, in the final packet row format. The resulting concatenated outputs from this block are MDB0, MDB1, MDB2 and MDB3 respectively, and their arrangement in terms of P0_S_i and P1_S_i are as follows:

MSB------------------------------------LSB

a) MDB0: P0_S_0 – P1_S_0 -- P0_S_1 – P1_S_1 (64-bits)
b) MDB0: P0_S_2 – P1_S_2 -- P0_S_3 – P1_S_3 (64-bits)
c) MDB0: P0_S_4 – P1_S_4 -- P0_S_5 – P1_S_5 (64-bits)
d) MDB0: P0_S_6 – P1_S_6 -- P0_S_7 – P1_S_7 (64-bits)

Note that every 16-bit P0_S_i/ P1_S_i further has an 8-bit real MSB and an 8-bit imaginary LSB breakdown and is transmitted in that order.

6) **MB_WR**: This subsystem is primarily responsible for generating write address and write enables in the correct sequence for memory write operation. It takes in Master_SYNC, SYNC from MERGE_POL block and user-input

N_CHAN. The Master_SYNC is used to clear all counters and latching logic. The MERGE_POL_SYNC is used to provide a trigger pulse/ enable for the counters, latches etc. The N_CHAN mainly serves as the 'select' input for several multiplexers that generate CHAN-specific outputs, ultimately resulting in the appropriate write-enables and write-addresses in the correct pattern.

7) **MB:** This is the primary memory bank of the design, that consists of 8 identical sub memory banks. The main inputs to the MB block are i) write-address, ii) 8 write-enables (one for each memory bank), iii) MDB0, MDB1, MDB2 and MDB3, iv) read-address and v) multiplexer select. In each of the 8 sub memory banks, the 64-bit MDB0/1/2/3 are first sliced to 2 32-bit words, so that it can be stored in the BRAMs. During transmission, the sliced words are concatenated back to ensure a standard packet format is being output. The outputs from MB_i, denoted by DX_i (i=0 to 7), are sent to the MB_RD block. The read-address and multiplexer select input to MB, are also generated in the MB_RD block.

8) **MB_RD**: This block is responsible for transmitting data from the memory banks to the eight 10Ge blocks in the standard packet format. It essentially comprises of 8 identical XFER_i (i=0 to 7) blocks (each corresponding to 1/8th of the total bandwidth), that continuously monitor write-enable signals to appropriately read out of the memory banks. This is ensured as the read operation takes place immediately after a memory write operation. The XFER_i also generates the read-address and multiplexer select codes, that serve as an input to the Memory Bank, at appropriate time instances. Most importantly, each XFER_i block transfers data in the standard packet to the corresponding 10Ge block for transmit, for that particular sub-band, in the following format:

a) Header: An 8-Byte header, comprising of a 1-Byte 'node-ID' (Most Significant Byte), and a 7-byte 'Frame Count' (Least Significant 7-Bytes), is appended.

b) Data Payload: The sub-band data, corresponding to N = (CHAN/8) of the total number of channels, is comprised of 1024 8-Byte words, and is arranged as follows:

S_0_P0_C0 -- S_0_P1_C0 -- S_0_P0_C1 -- S_0_P0_C1 (8-Bytes)

S_0_P0_C2 -- S_0_P1_C2 -- S_0_P0_C3 -- S_0_P1_C3 (8-Bytes)

.

.

S_0_P0_C(N-1) -- S_0_P1_C(N-1) -- S_0_P0_C(N-1) -- S_0_P1_C(N-1) (8-Bytes)

S_1_P0_C0 -- S_1_P1_C0 -- S_1_P0_C1 -- S_1_P0_C1 (8-Bytes)

S_1_P0_C2 -- S_1_P1_C2 -- S_1_P0_C3 -- S_1_P1_C3 (8-Bytes)

.

.

S_X_P0_C(N-1) -- S_X_P1_C(N-1) -- S_X_P0_C(N-1) -- S_X_P1_C(N-1) (8-Bytes)

Total Payload: 8192 Bytes

Here, S denotes the *complete-spectra*l index in the packet, X=8192/(4*N) correspond to the total number of spectra per packet; P0/P1 are the two polarization states, and C corresponds to channel index in the spectral subband, ranging from 0 to N-1. Also note that each S_P_C data can be further broken down to S_P_C_real and S_P_C_imaginary, and is transferred in that order.

c) Frame error count (FEC): An 8-Byte FEC (all 0s) is appended.

d) Inter-Frame Gaps (IG0 and IG1): Two interframe gaps IG0 and IG1, each 8 Bytes, at the end of each packet (alternating 1's and 0's).

Each XFER_i also generate strobe (STB_i) and end-of-frame (EOF_i) signals for the corresponding 10Ge_i blocks.

9) **tGX8**: This sub-system contains 8 identical 10Ge cores. Each 10Ge_i core is associated with a particular (1/8)th sub-band of the overall bandwidth, and thereby loads data (and STB_i/EOF_i) from the corresponding XFER_i, which is also operating for the same sub-band. The STB_i notifies the 10Ge_i block to start loading data to their transmit buffers. The EOF_i signal cause the 10Ge_i block to stop the load operation and start wrapping the contents to UDP packets, ready for transmission. We also have RUNT_i subsystems for each of the 8 10GE_i, that flushes out garbage data, once triggered by Master_SYNC pulse at the beginning of each observation, to ensure the transmit buffers are empty and ready for first legitimate packet stream.

In case of an overflow, the ‘tx_overflow’ output bit corresponding 10Ge_i core becomes TRUE, and remains TRUE, until the 10Ge cores are RESET. Each of 1-bit ‘tx_overflow’ bits are concatenated to form an 8-bit status signal, which is returned to the user via 32 bit ‘TXOF_STATUS’ SW registers, with an all 0’s first 24 bits, followed by the 8-bit status signal.

User-specified Variables: Here is a list of variables that the user can initialize, to assign/control the functioning of a few blocks. These variables are realized using SW registers in the actual hardware implementation:

1) **arm** : This variable is used to generate a one clock-cycle wide Master-SYNC pulse, coincident with the rising edge of the first 1PPS signal. A complete ‘arm’ command is performed by first writing a value of 1, followed by a 0.

2) **fftshift**: This variable determines if the stage outputs in the FFT block have to be divided by 2 (right shift). This is done on a stage-by-stage basis. Usually the value of FFT_SHIFT is set to AAAAAAAA(hex) for observations with weak signals and FFFFFFFF (hex) for strongly correlated signals to avoid overflows.

3**) ip_0 to ip_7**: Each 10Ge_i block set to the corresponding destination address IP_i.

4) **pt_0 to pt_7**: Each 10Ge_i block set to the corresponding port address PT_i.

5) **scale_p0/scale_p1**: Used to set the scaling factor for FFT outputs for both polarizations, and is input to the SCALE subsystem. The input values carry an Unsigned 32_16 data type.

6) **reset**: This variable is asserted if the 10GE transmit buffer enters to an overflow condition (This can be monitored using the TXOF_STATUS SW register). The assignment is similar to ‘arm’ (write a value of 1 and then 0).

7) **trig**: This variable is asserted to trigger capture in the ADC snapblocks. The assignment is similar to ‘arm’ (write a value of 1 and then 0).

8) **n_chan**: This variable is initialized according to the number of channels in the design. Note that this variable is *not* to be initialized with the actual number of channels. Instead, it should be assigned with the value “k”, where 2^k = CHAN.

9) **ssg_length**: This is used to denote the length of the switching signal (in terms of the number of spectral periods).

Real time monitoring subsystems: The following subsystems either allow real-time capture of signals within the ROACH-2, or provide us with Status signals to monitor the current state of the system. These sub-system can also be used for diagnostics and debugging of the firmware.

1) **adcsnap0/adcsnap1**: These snap blocks are used to capture 1024 concatenated 128-bit P0 and P1 ADC samples from the ASIAA_ADCs. This implies a capture of 16384 8-bit raw ADC samples from each of the polarizations. The capture begins once the user ‘triggers’ the blocks, by asserting the ‘trig’ SW register.

2) **Switching_Signal subsystem (SSB)**: This system returns the user with the current state the system is in, by either reading out of the SW register ‘status_out’, or monitoring the LED outputs of the ROACH (4:7).

Updates:

[1] DIBAS based raw ADC data-capture subsystem discontinued in future builds. Appropriate sections modified.
-- ASG (07/23/2015)

[2] Updated SW register names based on the new VPM Firmware-Manager instance naming convention.
-- ASG (07/23/2015)

[3] Updated Xilinx version to 14.7 .
-- ASG (07/23/2015)