LWA-SV Memo 1

LWA-SV F-engine firmware overview

D. Price   
28th January 2016

# Introduction

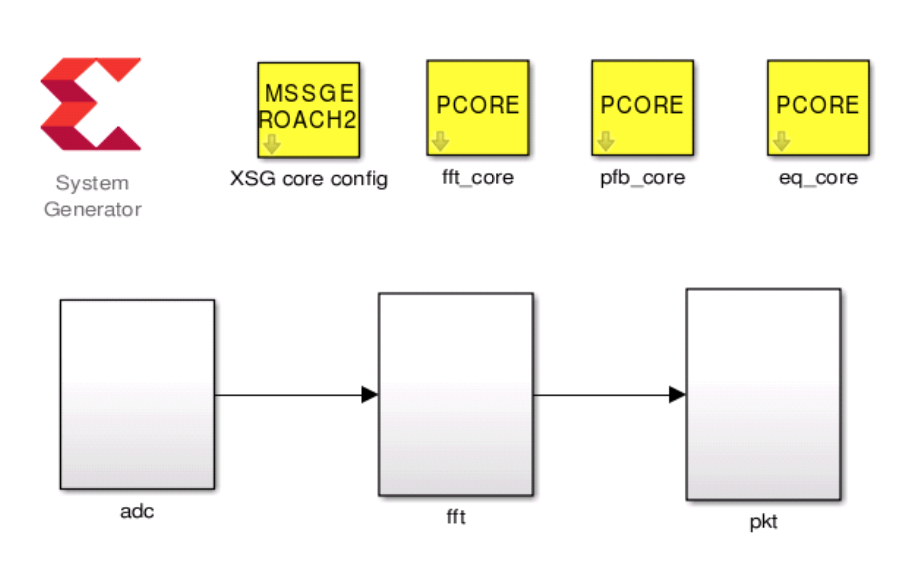
The LWA-SV digital signal processing system is known as the Advanced Data Processor (ADP). It consists of:

* 32x CASPER ADC16x250-8 digitizer cards (a total of 32x16=512 inputs).
* 16x CASPER ROACH2 FPGA processing boards (Xilinx Virtex-6 SX475T FPGA)
* Mellanox SX1024 10/40GbE switch
* 6x GPU servers (ASUS ESC4000 G3 server)

The digitization, channelization, and channel selection are done on the ROACH2 boards, the firmware of which is written using the CASPER / MATLAB / Simulink / Xilinx ISE toolflow[[1]](#footnote-1).

This memo gives an overview of LWA-SV firmware, by providing a walkthrough of the firmware’s Simulink diagram. It is intended that this document is read while clicking through the Simulink diagram, as it only provides overview of discrete blocks, and does not provide any insight into how the blocks fit together.

## Model walkthrough

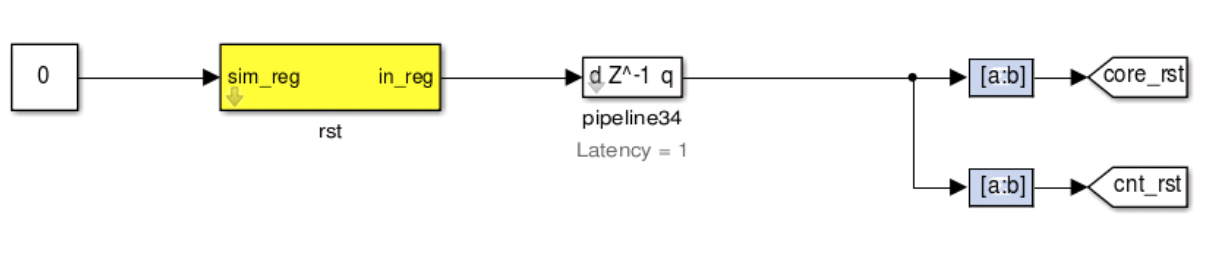


The model consists of three top-level blocks:

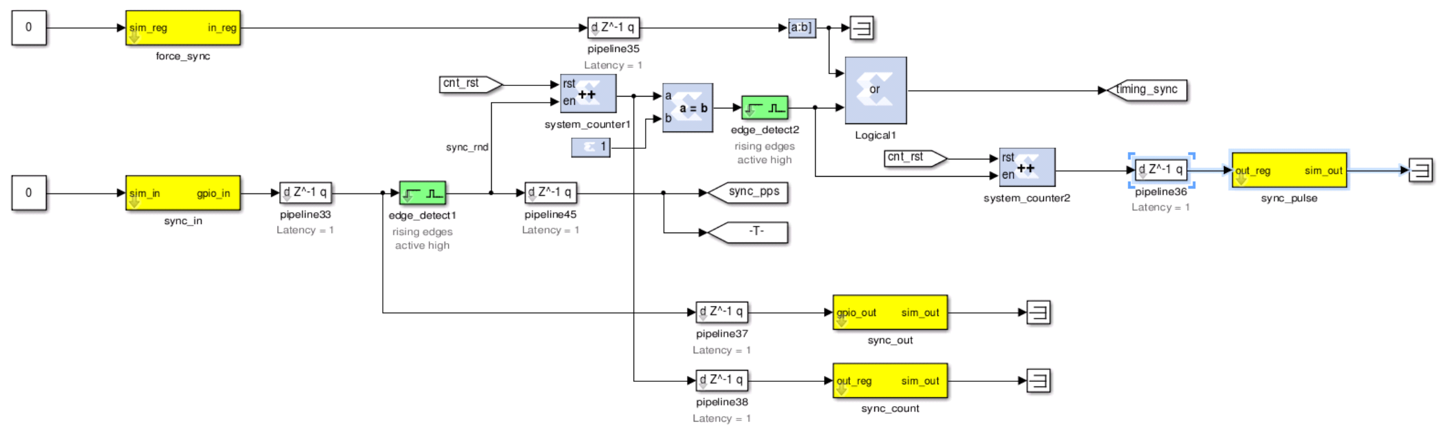
* **ADC** – this contains the ADC16x250 digitizer yellow block, and reset / synchronization pulse logic.
* **FFT** – this contains the polyphase filterbank implementation for each of 32 inputs, and the post-channelization 4-bit requantization logic.
* **PKT** – this contains 10 GbE Ethernet packetization logic, including channel selection.

## ADC top-level block

### Reset logic

 Main reset register *adc\_rst* provides a reset line for ‘cores’ (i.e. logic blocks), and counter values. The *adc\_rst* line should be set high (e.g. 0b11) and then low (0b00) to trigger a reset.

### Pulse-per-second and sync pulse



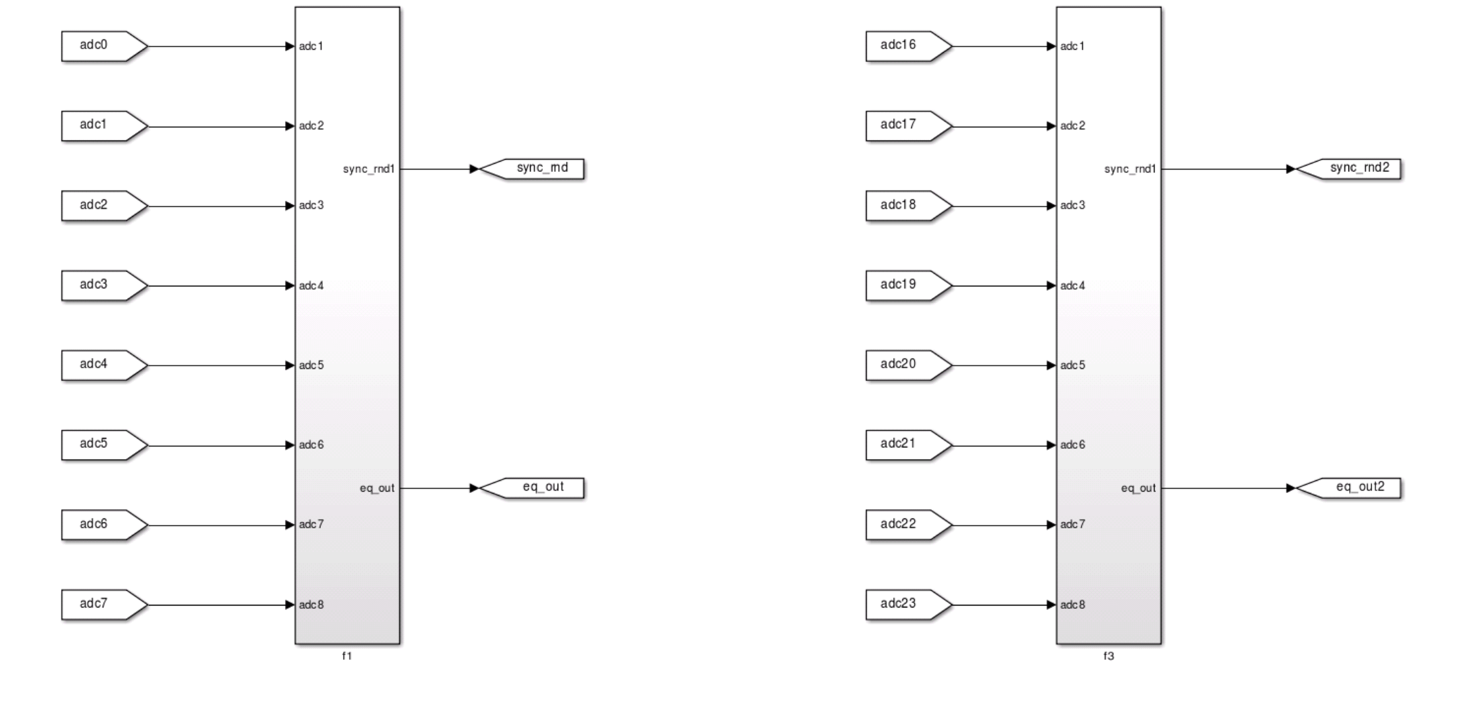
This logic is used to derive a ‘synchronization pulse’, that is required to reset internal logic of many CASPER blocks. Only one sync pulse will be generated and passed to the *timing\_sync* goto block, and a counter reset is required to trigger a new pulse to be propagated on *timing\_sync.*

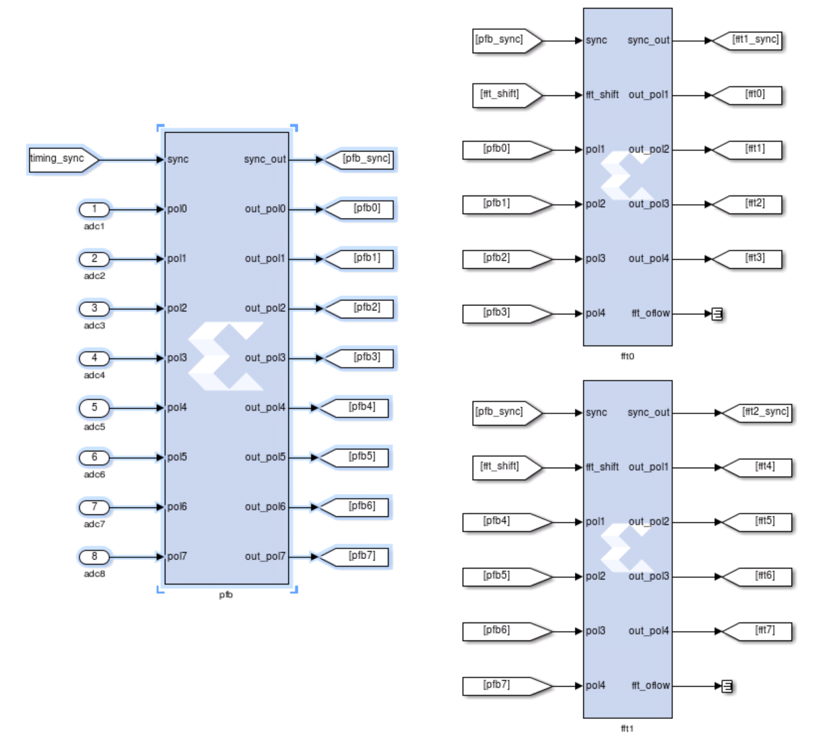
*adc\_sync\_in* is a GPIO connection that is connected to a pulse-per-second signal derived from GPS. A global goto *sync\_pps* is used in the packetizer to make sure values only change on a PPS, not mid-second.

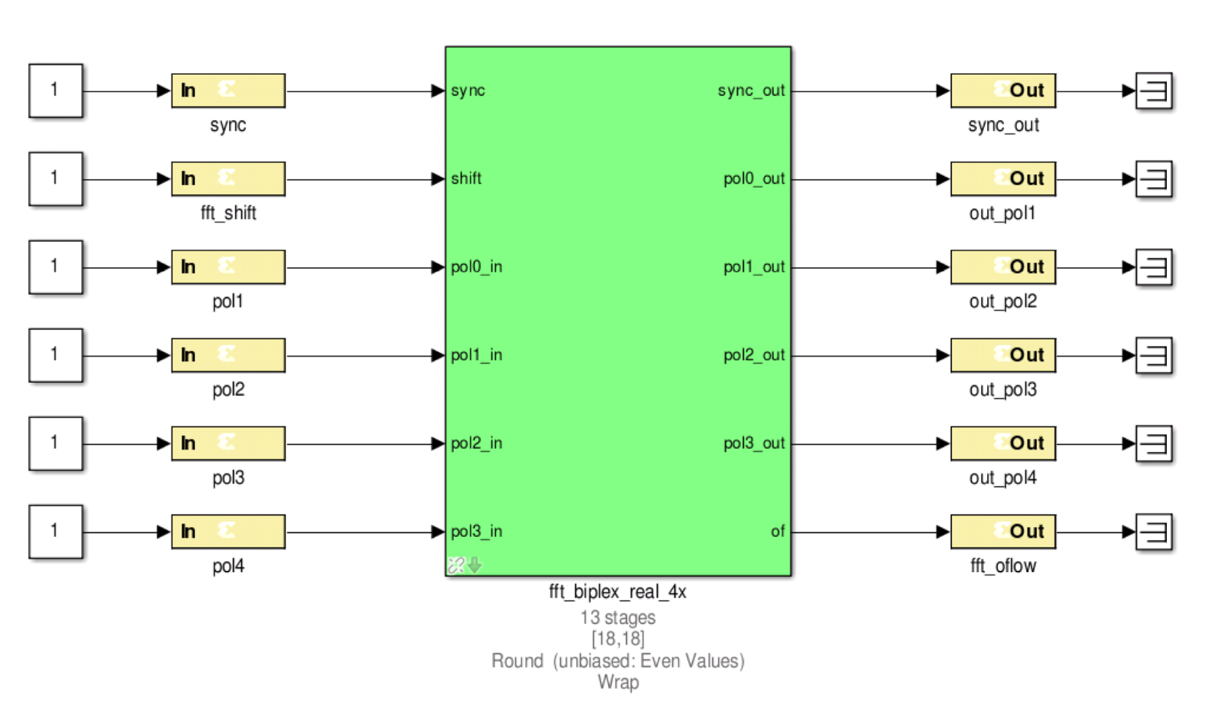
The *sync\_out*  GPIO propagates the sync pulse to the GPIO output (this is not used at LWA-SV). The *sync\_count* register stores a count of how many PPS have occurred since the last counter reset.  
The *sync\_pulse* register stores how many timing sync pulses have been sent.

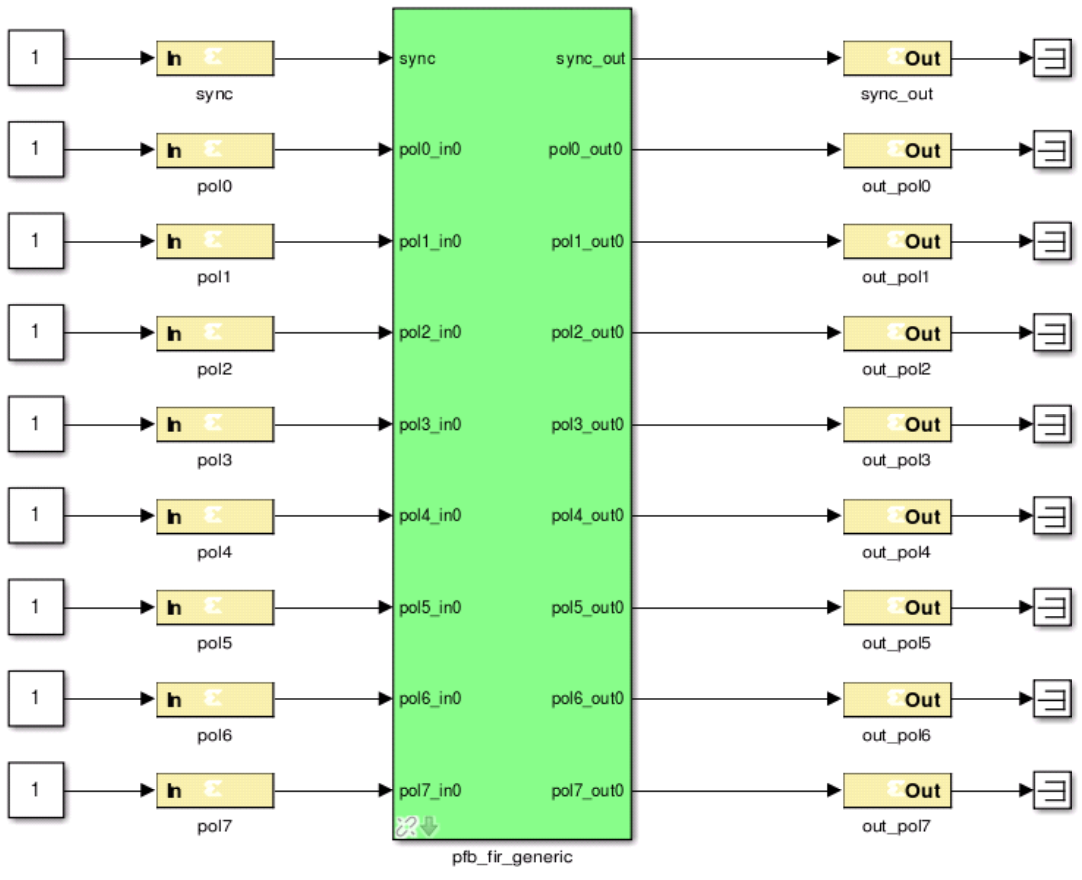
## FFT top-level block

### Sub-level PFB blocks

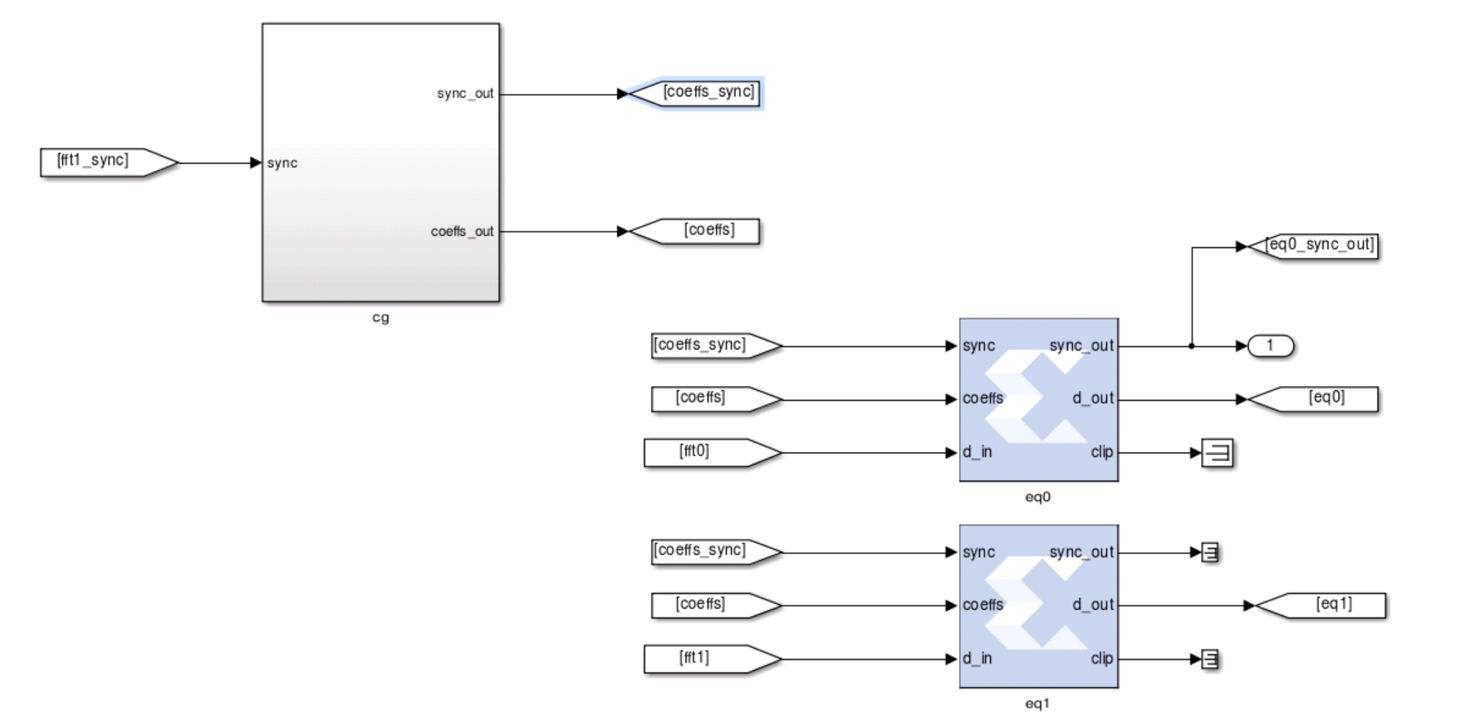
The PFB/FFT for the 32 input signals is split into 4 sub-blocks,  *f0* – *f3.* The output of each is a *sync\_rnd* signal, which is the propagated sync pulse, and *eq\_out,* which is all 8 requantized 4-bit input crammed together into a 64-bit word.

   
Inside each sub-block is 1x PFB FIR frontend, and 2x FFTs. Both of these are blackboxed.

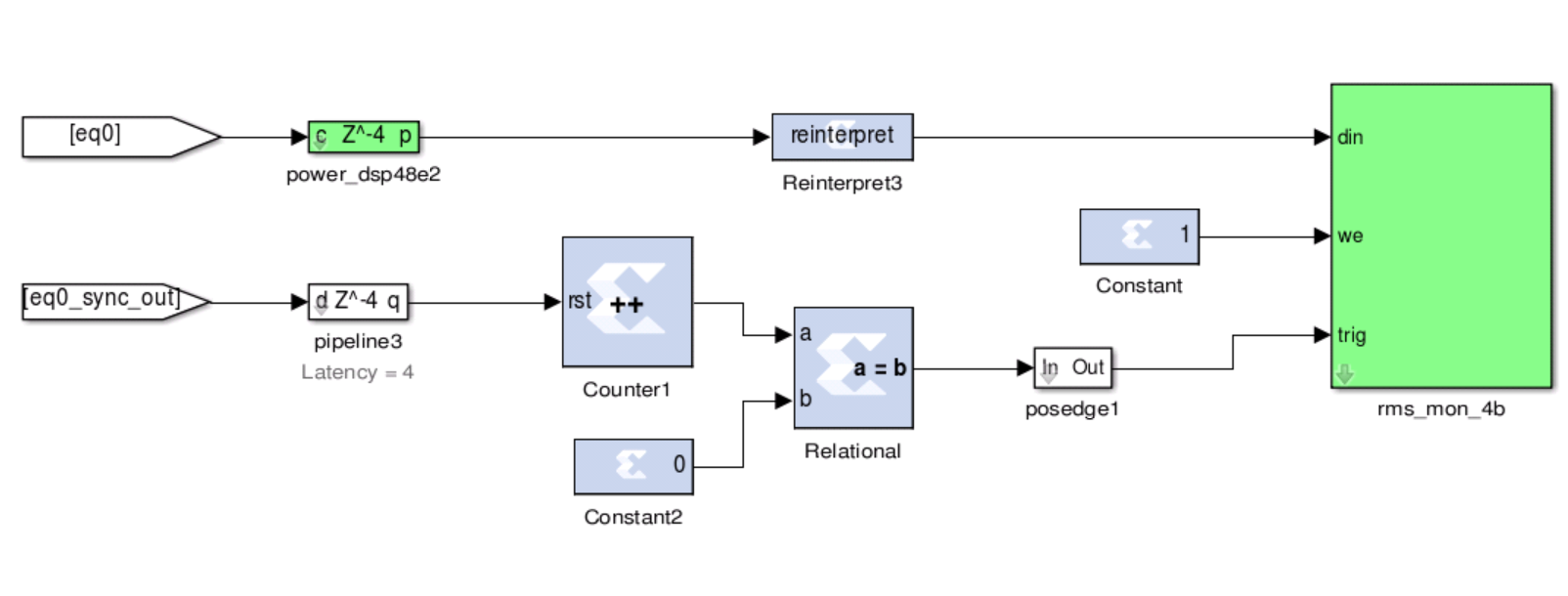
  
  
The core of each fft sub-block is of course an FFT. This is blackboxed (so in a separate simulink diagram), but has 13 stages (2^13 point FFT), and is a **biplex\_real\_4x** CASPER block.



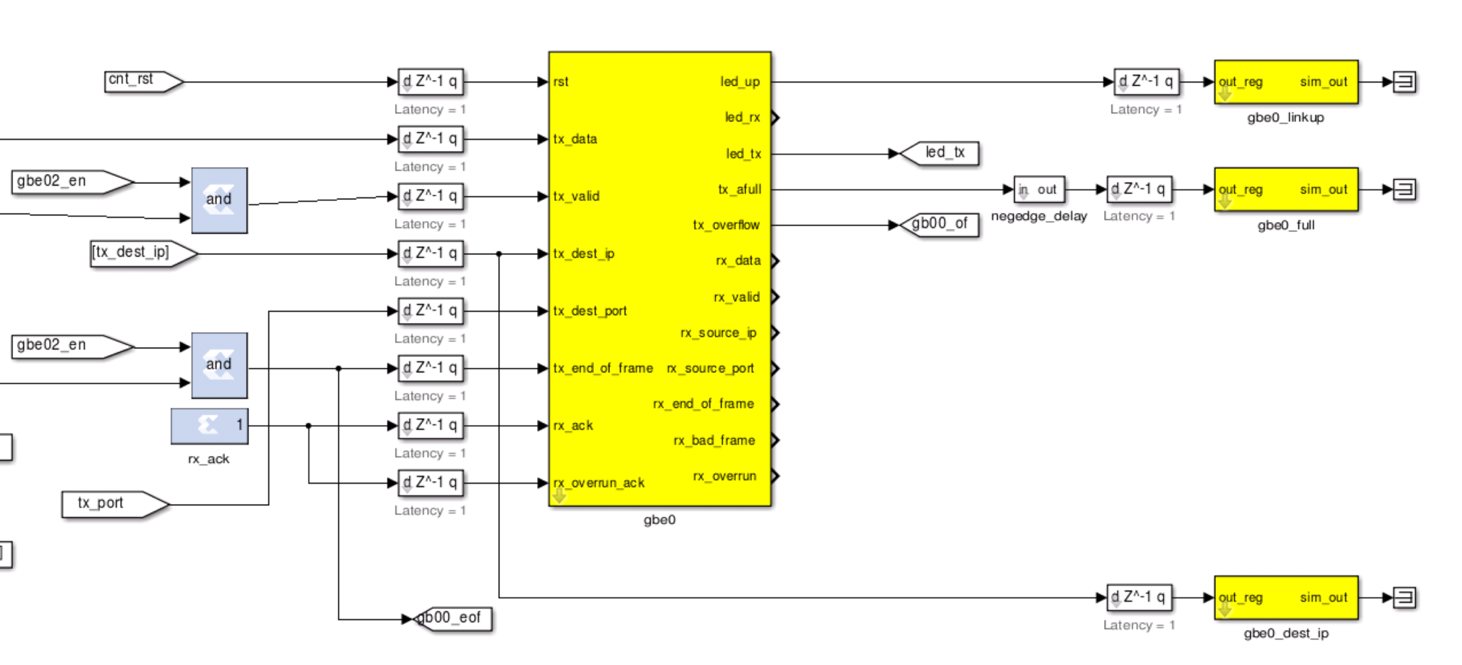
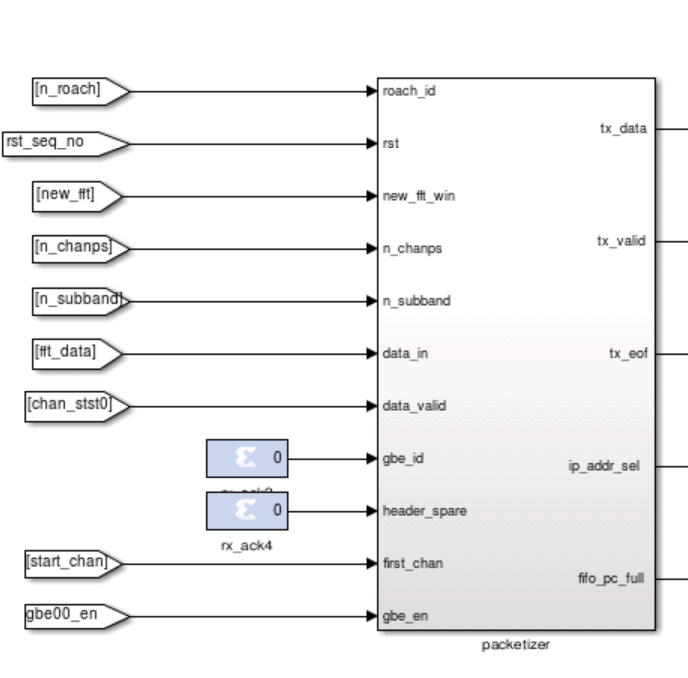
As the F-engine is a polyphase filterbank, the FFT is preceeded by an FIR low-pass prototype filter. This is also blackboxed.



Also under the FFT block is a 4-bit requantizer. This takes the 18\_re\_18\_im signal from the FFT and converts it down to 4\_re\_4\_im in the range [-7, 7]. In order to select the relevant bits, the signal is pre-multiplied on a per-channel basis, by a value written to a shared BRAM *fft\_f[0]\_cg\_bpass\_bram*.

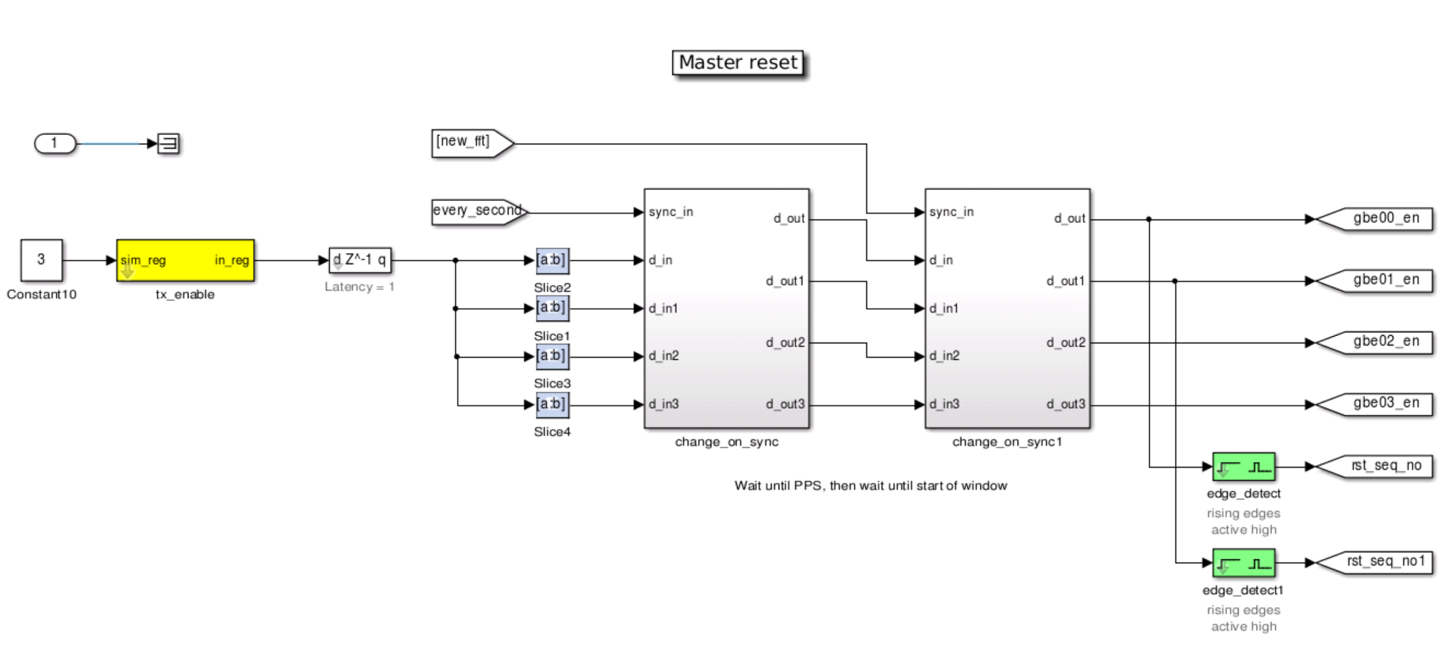
The final item in under a FFT block is a shared BRAM *fft\_f0\_rms\_mon\_4b,* which is essentially a 4-bit, single integration spectrometer. It allows the 4-bit quantization value to quickly tuned by providing an instantaneous snapshot of quantized values.

## PKT top-level block



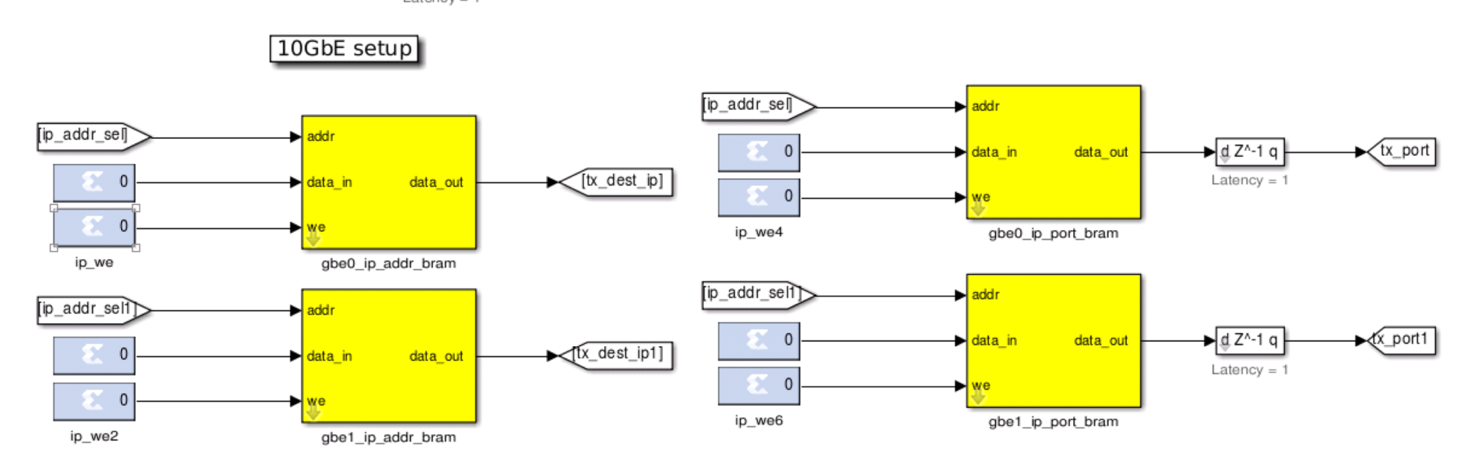
The packetizer selects channels to be sent out over 10GbE and

## /Users/dan/Desktop/Screen Shot 2016-01-28 at 1.36.48 PM.png/Users/dan/Desktop/Screen Shot 2016-01-28 at 1.36.56 PM.png

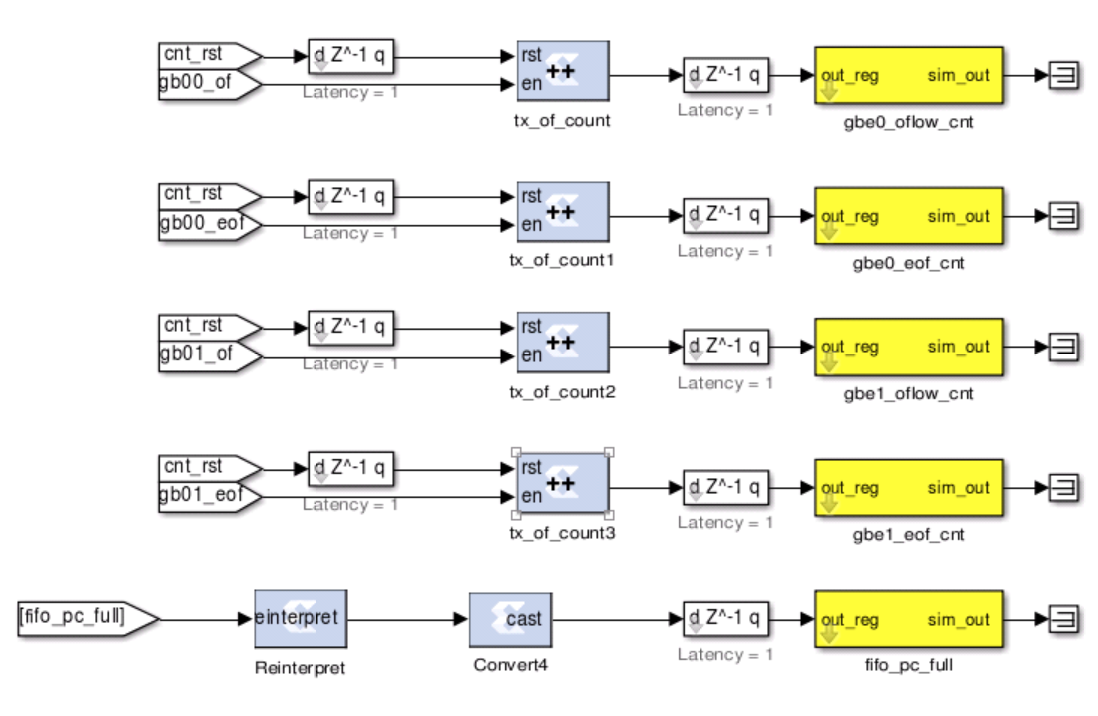


The packetizer has a whole bunch of user-configurable registers. The sync\_change and change\_on\_sync blocks are used so that values only change at the start of a PPS/FFT/sync window. The user-configurable registers are:

|  |  |
| --- | --- |
| Packetizer control registers |  |
| pkt\_roach\_id | Roach ID number. Should run 1-16, used to identify ROACH |
| pkt\_gbe0n\_chan\_per\_sub | Number of channels per subband for gbe0 packetizer.  Allowable values are 10-144 |
| pkt\_gbe1\_n\_chan\_per\_sub | Number of channels per subband for gbe1 packetizer.  Allowable values are 10-144 |
| pkt\_gbe0\_n\_subband | Number of subbands for gbe0 packetizer. Total number of channels sent will be n\_subband x n\_chan\_per\_sub.  Allowable values are 1-32 |
| pkt\_gbe1\_n\_subband | Number of subbands for gbe1 packetizer. Total number of channels sent will be n\_subband x n\_chan\_per\_sub.  Allowable values are 1-32 |
| pkt\_gbe0\_start\_chan | Start channel (lowest channel in range) for gbe0.  Allowable values 10-4000 |
| pkt\_gbe0\_start\_chan | Start channel (lowest channel in range) for gbe1.  Allowable values 10-4000 |
| pkt\_gbe0\_stop\_chan | Stop channel (highest channel in range) for gbe0.  Allowable values 20-4095 |
| pkt\_gbe1\_stop\_chan | Stop channel (highest channel in range) for gbe1.  Allowable values 20-4095 |
| pkt\_tx\_enable | Enable data flow. Controls data flow for both gbe0 and gbe1. LSB is for gbe0, LSB+1 is gbe1. To turn both on, write a value 0b11 = 3. To turn on gbe0 write 0b01=1 or for gbe1 0b10=2 |



|  |  |
| --- | --- |
| Packetizer control BRAMS |  |
| pkt\_gbe0\_ip\_addr\_bram pkt\_gbe1\_ip\_addr\_bram | List of IP address to send to. This should be the same length as the n\_subbands; each subband is sent to a corresponding IP in this list. |
| pkt\_gbe0\_ip\_port\_bram pkt\_gbe1\_ip\_port\_bram | List of IP ports to send to. This should be the same length as the ip\_addr\_bram. |



|  |  |
| --- | --- |
| Packetizer output registers |  |
| pkt\_gbe0\_oflow\_cnt pkt\_gbe1\_oflow\_cnt | This register will be >0 if there’s overflows happening in the 10GbE core. This probably means that your n\_subbands \* n\_chan\_per\_sub is too high. Only used in debugging. |
| pkt\_gbe0\_eof\_cnt pkt\_gbe1\_oeof\_cnt | Count of how many end of frame (EOF) have passed. More simply, the number of packets send out over the 10GbE link. Only used in debugging. |
| pkt\_fifo\_pc\_full | A fractional number showing what percent of the FIFO is full. Only used in debugging. |
| pkt\_gbe0\_linkup pkt\_gbe1\_linkup | A register that shows if the 10GbE core is configured and the link is up. Returns 1 if up, 0 if down. |
| pkt\_gbe0\_full pkt\_gbe1\_full | Returns 1 if the gbe0 or gbe1 blocks are filled. |

1. https://casper.berkeley.edu/wiki/Main\_Page [↑](#footnote-ref-1)