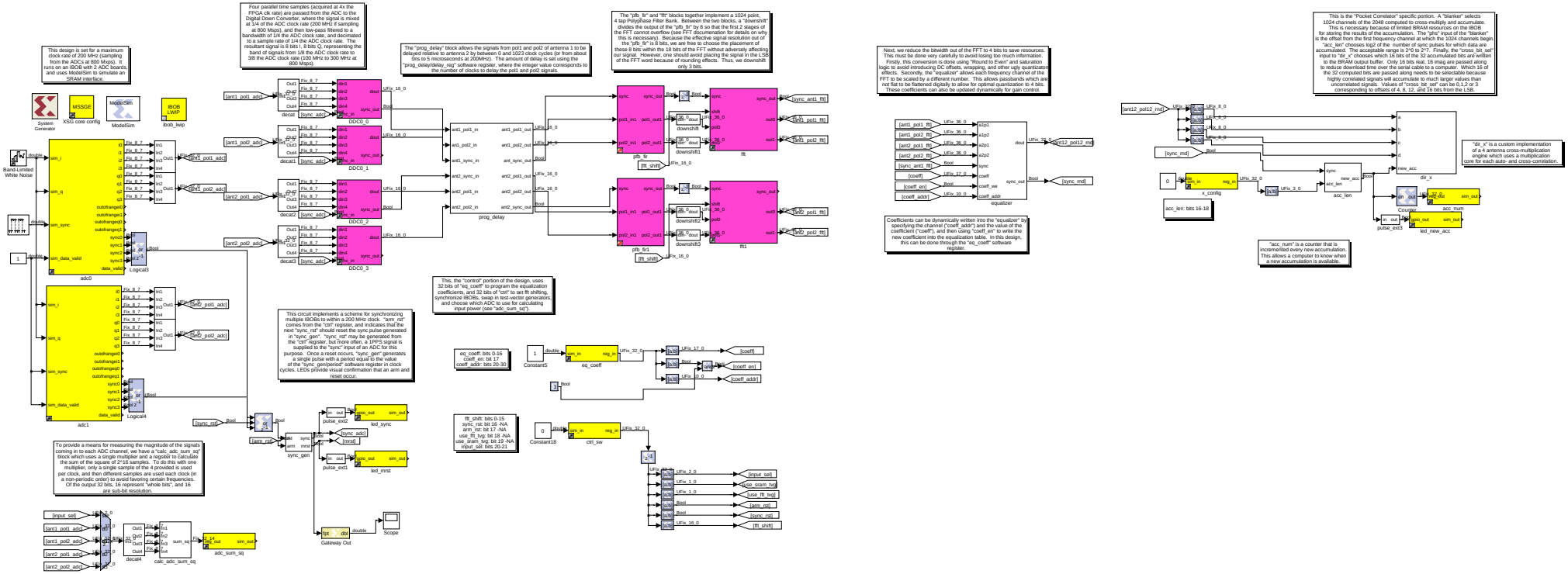
This design is set for a maximum clock rate of 200 MHz (sampling from the ADCs at 800 Msps). It runs on an IBOB with 2 ADC boards, and uses ModelSim to simulate an SRAM interface.
Four parallel time samples (acquired at 4x the FPGA clk rate) are passed from the ADC to the Digital Down Converter, where the signal is mixed at 1/4 of the ADC clock rate (200 MHz if sampling at 800 Msps), and then low-pass filtered to a bandwidth of 1/4 the ADC clock rate, and decimated to a sample rate of 1/4 the ADC clock rate. The resultant signal is 8 bits I, 8 bits Q, representing the band of signals from 1/8 the ADC clock rate to 3/8 the ADC clock rate (100 MHz to 300 MHz at 800 Msps).
The "prog_delay" block allows the signals from pol1 and pol2 of antenna 1 to be delayed relative to antenna 2 by between 0 and 1023 clock cycles (or from about 0ns to 5 microseconds at 200MHz). The amount of delay is set using the "prog_delay/delay_reg" software register, where the integer value corresponds to the number of clocks to delay the pol1 and pol2 signals.
The "pfb_fir" and "fft" blocks together implement a 1024 point, 4 tap PolyPhase Filter Bank. Between the two blocks, a "downshift" divides the output of the "pfb_fir" by 8 so that the first 2 stages of the FFT cannot overflow (see FFT documentation for details on why this is necessary). Because the effective signal resolution out of the "pfb_fir" is 8 bits, we are free to choose the placement of these 8 bits within the 18 bits of the FFT without adversely affecting our signal. However, one should avoid placing the signal in the LSB of the FFT word because of rounding effects. Thus, we downshift only 3 bits.
Next, we reduce the bitwidth out of the FFT to 4 bits to save resources. This must be done very carefully to avoid losing too much information. Firstly, this conversion is done using "Round to Even" and saturation logic to avoid introducing DC offsets, wrapping, and other ugly quantization effects. Secondly, the "equalizer" allows each frequency channel of the FFT to be scaled by a different number. This allows passbands which are not flat to be flattened digitally to allow for optimal quantization to 4 bits. These coefficients can also be updated dynamically for gain control.
This is the "Pocket Correlator" specific portion. A "blanker" selects 1024 channels of the 2048 computed to cross-multiply and accumulate. This is necessary because of limited BRAM resources on the IBOB for storing the results of the accumulation. The "phs" input of the "blanker" is the offset from the first frequency channel at which the 1024 channels begin. "acc_len" chooses log2 of the number of sync pulses for which data are accumulated. The acceptable range is 2^0 to 2^7. Finally, the "cross_bit_sel" input to "dir_x" chooses which 16 bits of the 32 accumulated bits are written to the BRAM output buffer. Only 16 bits real, 16 imag are passed along to reduce download time over the serial cable to a computer. Which 16 of the 32 computed bits are passed along needs to be selectable because highly correlated signals will accumulate to much larger values than uncorrelated signals. Values of "cross_bit_sel" can be 0,1,2 or 3 corresponding to offsets of 4, 8, 12, and 16 bits from the LSB.
"dir_x" is a custom implementation of a 4 antenna cross-multiplication engine which uses a multiplication core for each auto- and cross-correlation.
"acc_num" is a counter that is incremented every new accumulation. This allows a computer to know when a new accumulation is available.
acc_len: bits 16-18
This circuit implements a scheme for synchronizing multiple IBOBs to within a 200 MHz clock. "arm_rst" comes from the "ctrl" register, and indicates that the next "sync_rst" should reset the sync pulse generated in "sync_gen". "sync_rst" may be generated from the "ctrl" register, but more often, a 1PPS signal is supplied to the "sync" input of an ADC for this purpose. Once a reset occurs, "sync_gen" generates a single pulse with a period equal to the value of the "sync_gen/period" software register in clock cycles. LEDs provide visual confirmation that an arm and reset occur.
This, the "control" portion of the design, uses 32 bits of "eq_coeff" to program the equalization coefficients, and 32 bits of "ctrl" to set fft shifting, synchronize IBOBs, swap in test-vector generators, and choose which ADC to use for calculating input power (see "adc_sum_sq").
eq_coeff: bits 0-16
coeff_en: bit 17
coeff_addr: bits 20-30
fft_shift: bits 0-15
sync_rst: bit 16 -NA
arm_rst: bit 17 -NA
use_fft_tvg: bit 18 -NA
use_sram_tvg: bit 19 -NA
input_sel: bits 20-21
To provide a means for measuring the magnitude of the signals coming in to each ADC channel, we have a "calc_adc_sum_sq" block which uses a single multiplier and a regsiblr to calculate the sum of the square of 2^16 samples. To do this with one multiplier, only a single sample of the 4 provided is used per clock, and then different samples are used each clock (in a non-periodic order) to avoid favoring certain frequencies. Of the output 32 bits, 16 represent "whole bits", and 16 are sub-bit resolution.
System Generator
XSG core config
ModelSim
IBOB LWIP
ibob_lwip
Band-Limited White Noise
adc0
adc1
DDC0_0
DDC0_1
DDC0_2
DDC0_3
prog_delay
pfb_fir
fft
pfb_fir1
fft1
downshift
downshift1
downshift2
downshift3
equalizer
x_config
acc_len
dir_x
Counter
acc_num
led_new_acc
eq_coeff
ctrl_sw
sync_gen
led_sync
led_mrst
Gateway Out
Scope
calc_adc_sum_sq
adc_sum_sq