Correlator Development Plans

Aaron Parsons

April, 2006

Background

This document is a status report on the development of an 8-station FX correlator using BWRC's BEE2 hardware (which includes the BEE2, iBOB, and iBOB-ADC board) powered by Xilinx Virtex 2-Pro FPGAs. Designs for these FPGAs are being constructed using the Matlab/Simulink/System Generator toolset. Our libraries include designs for a biplex-pipelined FFT, an FIR front-end for implementing Polyphase Filter Banks, frequency-domain cross-multiplication (X engines), and matrix transpositions in BRAM and DRAM.
We are currently in the process in implementing an 8-station correlator that will be a useful stepping-stone for the development of the **Allen Telescope Array** and for Don Backer's **PAPER Array** in Greenbank.

Phase I: 8 Antennas, 256 Channels, 4 bits, 2 Stokes

To make this phase of correlator development the simplest possible system with scientific functionality, we have partitioned the digital down-converter (DDC), polyphase filter bank (PFB), time-frequency corner turn (CT), barrel switcher (SW), X engine (X), and vector accumulator (VACC) among 4 iBOBs and 1 BEE2 as follows:

- The iBOB contains only the DDC. This simplifies the problem of synchronizing multiple boards together by keeping data in the time-domain as it passes through the variable latency XAUI links to the BEE2 board. Thus, the unknown latency (which is a static after powering the correlator) will appear as a fixed phase offset between antennas during an observing run. This offset may be calibrated away.

- The BEE2 User FPGA contains the PFB and the CT, with the CT being implemented in on-FPGA BRAM resources. On-chip data reordering sets a lower limit on the output bandwidth from an X engine into a long-term accumulation buffer. A single FPGA can only hold around 2 dual-polarization PFBs, so this architectural choice limits the number of antennas to 8.

- The BEE2 Ctrl FPGA contains the SW, X, and VACC. The implementation of the entire X engine on 1 FPGA simplifies data routing and switching, but multiplier and slice resources limit the number of bits and Stokes parameters which may be carried through the X engine. The VACC is implemented in on-chip BRAM, which limits the number of frequency channels/Stokes parameters which may be accumulated.

Benefits and Limitations

The primary benefits of the Phase I correlator were its cut-back reliance on hardware infrastructure and its simplistic synchronization scheme, both of which helped shorten development time.
This version of the correlator was most heavily limited by the resources available implementing CMACs (complex multiply and accumulate) in the X Engine. The number of CMACs needed for the X component of an N_{ant} array is:

$$N_{cmac} = floor\left(\frac{N_{ant}}{2} + 1\right) \times N_{ant} \times N_{Stokes}$$

(see Appendices A and B for details). The limited number of multipliers on a chip (see Appendix B), forced us to branch into two different implementations of this correlator: one with 3 bit samples and all Stokes parameters, and one with 4 bit samples and half the Stokes parameters.

Phase II: 16 Antennas, 4096 Channels, 4 bits, 4 Stokes

This phase of correlator development will overcome the channel and antenna limitations of Phase I by taking advantage of off-FPGA SRAM and DRAM, and by adopting a more complicated synchronization scheme which allows a more advantageous partitioning of the signal processing logic.

- The IBOB will now contain the DDC, PFB, and the CT, with the CT being implemented in off-chip SRAM. Combined with a VACC implemented in DRAM, this will expand the number of channels to 4096. Implementing the CT in SRAM allows the lengthening the short-term accumulation window to 128 in the X engine, which is necessary to meet the data-rate bottleneck ($\sim$ 4 GB/s for sustained bidirectional flow) into DRAM. Since 2 dual-polarization PFBs need to be transposed, with each polarization represented by a 4-bit real/4-bit imaginary complex number, the equation for the size of this corner-turn is:

  $$N_{BRAM} = N_{freq} \times (2\ antennas) \times (2\ polarizations) \times (2 * \frac{bits}{sample}) \times W_{acc} \frac{1BRAM}{18kbits}$$

  where W_{acc} is the number of spectra which are buffered for short-term accumulation in the X Engine. On the IBOB, In practice, $N_{BRAM} \sim 200$, and this would limit N_{freq} to 512.

- The BEE2 User FPGAs will each contain one fourth of the SW and the X. In order to switch data to X engines on adjacent User FPGAs, interchip connections on the BEE2 will be used. To switch data to X engines on the diametrically opposite User FPGA, a connecting XAUI link will be used. Splitting the X engine across multiple FPGAs will allow us to move to 16 antennas and allow us to support more bits and all Stokes parameters through the X engine.

- The BEE2 Ctrl FPGA will contain the VACC, which will make use of off-chip DRAM to support the accumulation of many frequency channels for many antennas.

Benefits and Limitations

The simple point-to-point data communication between boards allows for the construction of a moderate sized correlator without delving into the headache of packetized protocols. This iteration

on correlator development expands the number of channels which may be supported close to the limit of what is scientifically valuable, and expands the number of antennas which may be supported.
The size of this correlator is most immediately limited by the number of multipliers and slices available on a BEE2 board. Less immediate, but looming, are limitations on the bandwidth between chips and the number of XAUI cables which may be physically plugged into a BEE2 board. Overcoming these limits requires splitting the X engine across multiple boards, and this will require a switch to route data from a single antenna to multiple engines.

Phase III: Many Antennas, 4096 Channels, ? bits, 4 Stokes

This final phase of correlator development will overcome the single X engine board limitation by implementing a packetized 10 Gb Ethernet protocol over XAUI links and moving the SW from the BEE2 into a network switch. In this case, the signal processing is divided up as follows:

- The IBOB design remains the same as in Phase II, except for packetization.
- With the SW being implemented in a network switch, each BEE2 User FPGA need only implement as many X engines as fit. There is no more need for communication between User FPGAs, and X Engine results are passed to the Ctrl FPGA.
- The BEE2 Ctrl FPGA will remain essentially identical to the design in Phase II.

Benefits and Limitations

This architecture will be scalable to essentially any number of antennas. The number of channels is capped at what can fit in an IBOB, and expansion beyond then will require the design of an IBOB upgrade.

Future Correlator Techniques/Technologies

In order to broaden the applicability of our correlator architecture, there are several techniques/technologies we will need to implement. Some of these may also be necessary for the Phase III (packet switched) correlator.

1. Trading reduced bandwidth for fewer X engines. This solution is relatively straightforward to implement. Supposing that we are only interested in some fraction of the output channels from a PFB, we can place the channels we want into a FIFO as they emerge from the CT. When reading from the FIFO, we alternate reading some number of channel blocks (each block has length W_{acc}), with blocks of zeros such that after the barrel switch, all of the zeros get routed to one or more X engines, and none of the data gets routed to these X engines. These X engines may then simply be omitted. This allows us to remove one X engine for every $\frac{N_{freq}}{N_{ant}}$ fraction of the band we throw out.
2. Wideband correlation. With our wideband FFT (capable of handling bandwidths which are powers of 2 times the FPGA clock rate), we have all the tools to make a wideband correlator. If B is the number of streams (samples which show up simultaneously), then we simply

implement B CTs for each FFT (1 for each stream), B barrel switchers (one to switch all the stream 1s from of the FFTs, one for the 2s, etc.), and then $B \times N_{ant}$ X engines (one for each stream). Thereafter, all that remains is to unscramble the post-correlation frequency channels in software.

3. Running X engines at a different (higher) clock rate than the F engines. This is one of the more difficult technologies to develop, as it will require the creation of an "X Engine Unscrambler" block which outputs all the cross-correlations for a frequency channel contiguously (the current X engine has some interleaving of results for different frequency channels, see Appendix A). A precursor of this block has already been implemented by Pat Crescini.

   Once we have this block, we may proceed as follows: an X engine correlates results from a $N_{ant} \times W_{acc}$ window of samples. Data from adjacent windows does not influence the results from a given window. Thus, we may precede each X engine with a FIFO of length $2 \times N_{ant} \times W_{acc}$. When this FIFO has at least $N_{ant} \times W_{acc}$ samples buffered, and an X engine window is beginning, we may send 1 window of data through the X engine. If a X engine window is beginning but the FIFO does not have all the samples for that window, no data is read from the FIFO, and the data valid signal for the output data from this window should be masked before it enters the VACC.

   It will be important in this case that the accumlation enable signal for the VACC be generated by counting enable signals, and not by simply counting clocks (which is done in the current Phase I correlator).

4. A number of antennas which is not a power of 2. The basis of this technology is essentially the same as for running X and F engines on different clocks: FIFOs need to be added to the inputs of X engines. This will enable us to remove some (small) number of frequency channels from a spectrum such that the number of resulting channels is divisible by the number of antennas. By throwing out channels, we are essentially reducing the sample rate of the F engine output, and thus we need a FIFO to absorb timing differences and align data with X engine windows (as per the previous discussion).

Appendix A: X Engine Data Output Order

1^{st}.	0×0	$\langle 0 \times N \rangle$	$\langle 0 \times (N-1) \rangle$	$\langle 0 \times (N-2) \rangle$	...	$\rightarrow$
2^{nd}.	1×1	0×1	$\langle 1 \times N \rangle$	$\langle 1 \times (N-1) \rangle$	...	$\rightarrow$
3^{rd}.	2×2	1×2	0×2	$\langle 2 \times N \rangle$	...	$\rightarrow$
4^{th}.	3×3	2×3	1×3	0×3	...	$\rightarrow$
5^{th}.	4×4	3×4	2×4	1×4	...	$\rightarrow$
...	...	...	...	...	...	$\rightarrow$

- Bracketed samples represent products from the previous frequency slice.
- Data shifts out of the right, starting with the top row, and each row output corresponds to a data valid pulse of length $floor\left(\frac{N}{2}+1\right)$. Thus, the total number of output results for a single frequency slices is:

$$N_{results} = N_{ant} \times floor\left(\frac{N}{2}+1\right)$$

- If N is even, then the final column only has valid results for the 2nd $\frac{N}{2}$ samples out.

Appendix B: FPGA Resources

Chip	FPGA type	MULTs/Used	BRAMs/Used	LUTs/Used	Flops/Used
iBOB	2VP50-7	232/0	232/34	50,000/1,343	50,000/1,311
BEE2 (User)	2VP70-7	328/0	328/34	70,000/1,024	70,000/1,157
BEE2 (Ctrl)	2VP70-7	328/0	328/76	70,000/7,019	70,000/7,456

Table 1: Available FPGA Resources/Resources Used in Base System

A single slice has 2 LUTs and 2 FLOPs. Generally, using LUTs and FLOPs in a single slice for unrelated logic sharply reduces the clock speed of a chip. Furthermore, using more that 90% of the slices in a chip incurs a high penalty for clock speed. Thus, the effective resources on a chip are significantly less than the numbers above indicate.

Design	MULTs	BRAMs	LUTs	Flops
4 bit CMULT	4	0	18	34
4 bit CMULT	3	0	47	47
4 bit CMULT	0	0	82	114
3 bit CMULT	0	0	42	94
3 bit CMULT	1	0	44	43
12 bit ACC	0	0	28	52

Table 2: CMULT and ACC Resources

Design	MULTs	BRAM	LUTs	Flops
4 bit, 2 Stokes (3/8 using 4 MULTs + 5/8 slices)	120	0	9,120	15,040
3 bit, 4 Stokes (8x using 1 MULT)	160	0	16,000	23,520
4 bit, 4 Stokes (3/8 using 4 MULTs + 5/8 slices)	240	0	18,240	30,080

Table 3: Total X Engine Resources

Appendix C: Detailed Development Steps for Phase II

1. Develop an SRAM interface which allows an effective read-modify-write. That is, overall we should be able to average 1 sample written and 1 sample read per clock.

2. Edit the VACC such that the accumulation FIFO is split into 2 small FIFOs which interface to a DRAM controller. When the input FIFO is over a threshold, data is written into DRAM, and when the output FIFO is under a threshold, data is read from DRAM.

3. Move the PFB and CT into the iBOB. Each IBOB will have onboard sync generation which is reset by 1PPS when an arm signal is set. This arm signal will come as an out-of-band signal from the User FPGA, and will have been ultimately generated on the Ctrl FPGA. Otherwise, the XAUI sync logic should remain very similar to what has been implemented in Phase I. Make the User FPGA a pass-through to the Ctrl FPGA. This design should remain functionally equivalent to the Phase I correlator, except for the number of PFB taps and the DDC filter order.

4. Move the CT into SRAM, and the VACC into DRAM. This will be functionally similar to Phase I, but will have a longer short-term integration to meet the bandwidth into DRAM (CT accumulation should be ~ 128).

5. Raise the number of channels to 4096.

6. Move the SW from the Ctrl FPGA to the 4 User FPGAs. Each User FPGA takes 2 antennas from an IBOB. Add a XAUI link between each corner FPGA and its opposite partner. The User FPGA will now switch antenna channels to each of the other corner chips, and after this switching, each will pass their data unmodified to the Ctrl FPGA. Don't forget to pass handshaking signals to the Ctrl FPGA, and then back to the User FPGAs. This stage will be functionally equivalent to the previous stage.

7. Raise the number of antennas to 16.