

# Signal Processing at 250 MHz Using High-Performance FPGA's

Brian Von Herzen

**Abstract**—This paper describes an application in high-performance signal processing using reconfigurable computing engines: a 250-MHz cross correlator for radio astronomy. Experimental results indicate that complementary metal-oxide-semiconductor (CMOS) field programmable gate arrays (FPGA's) can perform useful computation at 250 MHz. The notion of an "event horizon" for FPGA's leads to clear design constraints for high-speed application developers, and can be applied to a variety of real-time signal processing algorithms. Recent estimates indicate that higher performance FPGA's available early in 1998 can attain speeds of over 300 MHz using 20% fewer logic elements than current designs. The results of this design work provide important clues on how to improve FPGA architectures for signal processing at hundreds of MHz. Direct routing channels between logic elements can significantly increase performance. Routing architectures with four-way symmetry allow for rotations and reflections of subcircuits needed for optimal packing density. Experimental results indicate that clock buffering often limits the top speed of the FPGA. Wave pipelining of clock distribution network may improve FPGA performance.

**Index Terms**—Correlators, event horizon, field programmable gate array, manual partitioning and placement, programmable logic, real-time signal processing.

## I. RECONFIGURABLE COMPUTERS

**F**IELD-PROGRAMMABLE gate arrays (FPGA's) can provide a useful platform for high-performance computing and real-time interactive signal processing [1]. These arrays perform well on real-time computations because they provide the speed of dedicated circuitry while retaining the flexibility of a programmable system. Reconfiguring allows for incremental optimization and improvement of hardware much like software development techniques. Because each element in an FPGA performs a dedicated task, application developers can readily design each circuit in the system for a specified performance, ideal for real-time signal processing, which requires that data flow through the system at a specified rate.

This paper will describe a real-time application in radio astronomy that makes use of the fastest FPGA's available, and performs a computation at speeds heretofore only achievable in custom silicon [2]. The reconfigurable computer retains the flexibility to implement different performance tradeoffs, allowing for large arrays of simple computations or short arrays

of complex computations. Reconfiguring the computer on demand provides a unique flexibility that has not existed before.

## II. CORRELATION SPECTROMETERS

Radio astronomers analyze spectra using high-performance real-time computers [3]. High-frequency radio astronomy, particularly millimeter-wave and submillimeter astronomy, utilizes wide-band spectrometers with at least 1–2 GHz of spectrometer bandwidth. Weak astronomical signals require full-parallel integration on all of the channels to detect measurable signals. Therefore, scanning spectrometers do not work for this application. Acoustooptic spectrometers can achieve the broad bandwidth, but become problematic in large arrays and for space-borne applications where adjustments of the analog elements become very difficult.

Another alternative uses a parallel digital correlator that computes the autocorrelation function of the incoming base-band signal [2]. A digitizer quantizes the signal at a resolution of one, two, or three bits [4]. A 1:16 time-division demultiplexer reduces the data rate from a Nyquist sampling rate of 4 Gs/s down to 250 Ms/s, producing 16 parallel data streams of 250 Ms/s. The streams pass into an array of cross correlators that correlate every stream with every other 250 MHz stream. Digital integrators accumulate the cross-correlation results for periods from  $10^6$  to  $10^{12}$  samples, and the integration results pass to a microprocessor for further integration. The processor reassembles the autocorrelation of the 2 GHz input signal from the array of cross-correlation results and computes the Fourier transform of the correlation, producing the spectrum of the 2 GHz signal.

In this paper, we will focus on the 250 MHz cross-correlation portion of the algorithm, the heart of the real-time computation. This building block serves as the basic element in a large array of spectrometers for the Caltech Submillimeter Observatory [5] and the James Clerk Maxwell Telescopes, Mauna Kea, HI [6].

### A. Cross-Correlator Architecture

Fig. 1 shows the basic architecture of a single cross correlator. Two 2-bit digital signals enter the correlator, called the prompt and delayed signal. Each lag of the correlator delays the delayed signal by one more clock than the prompt signal, hence the naming convention. A hardware multiplier at each lag computes the product of the prompt data and the delayed data, and offsets and rounds the result to 3 bits. An accumulator integrates the rounded products and passes its

Manuscript received March 25, 1997; revised November 1, 1997. This work was supported by The Caltech Submillimeter Observatory, and the Joint Astronomy Center and Xilinx, Inc.

The author is with Rapid Prototypes, Inc., Carson City, NV 89701 USA. Publisher Item Identifier S 1063-8210(98)02952-7.

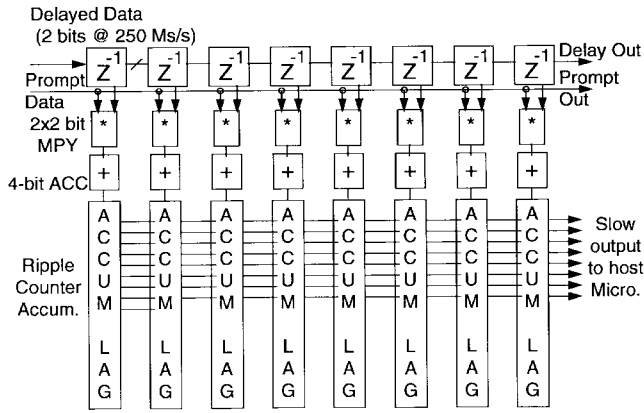


Fig. 1. Block diagram of a cross correlator.

TABLE I

MODIFIED AND OFFSET MULTIPLICATION TABLE FOR TWO-BIT CORRELATION INPUT SIGNALS. THE MULTIPLIER ROUNDS AND OFFSETS THE PRODUCTS TO PRODUCE THREE PRODUCT BITS INSTEAD OF FOUR, ALLOWING 3-BIT UNSIGNED ADDITION FURTHER DOWN THE PIPELINE

2 x 2 bit mpy	Multi- plicand	-3	-1	+1	+3
Multi- plier	Signed - Magn.	1 1	1 0	0 0	0 1
-3	1 1	6	4	2	0
-1	1 0	4	3	3	2
+1	0 0	2	3	3	4
+3	0 1	0	2	4	6

carry bit to a ripple counter acting as an accumulator. The ripple counter accumulates for integration times approaching 10 billion samples for astronomical applications, whereupon the results pass to a host computer, the counters reset, and the integration process repeats. This computation occurs at each lag in the correlator. The prompt and data signals emerge from the right side of the correlator to permit daisy-chaining of the correlator chips to form longer correlators.

B. Design of an Individual Correlator Lag

Fig. 2 shows the architecture of an individual correlator lag. Delayed and prompt data enter the lag on the west side and pass through registers on the rising edge of the clock. Combinational logic computes a three-bit rounded product using the values listed in Table I. In the signed-magnitude number representation, the data values correspond as follows: (11) = -3, (10) = -1, (00) = +1, and (01) = +3. Normally this would produce products from -9 to +9. We offset these products by +9 and divide by three to get the range from zero to six. Then we round the four central entries in the table to the value three to get integer values. The resulting entries appear in Table I.

Table I produces 3-bit products that have LSB's of the inner products rounded, with all the products offset to produce positive results. The offset allows the logic to have unsigned adders, accumulators, and up-counters. Later a micro-processor eliminates the offset by tracking the total number of samples and subtracting the number of samples times the fixed offset of three from the correlator results.

TABLE II  
TIMING BUDGET AT 250 MHz FROM ONE SYNCHRONOUS REGISTER TO ANOTHER. THE DI PIN BYPASSES THE LOOK-UP TABLE (LUT), PROVIDING A LONGER MAXIMUM ALLOWED WIRE DELAY

Timespec	through LUT	via DI pin
Total cycle time	4.0	4.0
On-chip clock skew	0.1	0.1
Clock to output time	1.3	1.3
Setup time	1.5	1.0
Maximum wire delay	1.1	1.6

The product goes to a 4-bit accumulator whose carry output controls a ripple counter for integration. The correlator runs for a predetermined number of cycles, after which the integration results go to a microcontroller for low-bandwidth processing.

This basic algorithm repeats for each lag in the correlator array. The next two sections describe the optimizations to the architecture needed to obtain the performance objectives of 250 MHz real-time throughput with the correlator.

III. HIGH-SPEED FPGA'S

FPGA technology frequently uses lookup tables based on static memory coupled with synchronizing registers. For this application, we used the XC3195-09 chip from Xilinx [7]. This architecture utilizes two 4-bit lookup tables and two flip-flop registers per configurable logic block (CLB). These chips provide ample resources for small-scale pipelining, the key to achieving high throughput in FPGA's.

Two approaches to high-speed FPGA design include top-down system design and bottom-up library element design. Here we use a bottom-up methodology where we set the target clock frequency in advance, and then design each element to meet the performance objectives from the start. These library elements connect using standard synchronous pipelining as long as the connections between elements remain short enough to stay within the synchronous timing window. We chose to design at 250 MHz based on rough performance estimates and the constraint that the clock frequency must divide the sampling rate of 4-GHz by a power of two. In addition, we had completed a full-custom design at 250 MHz using 1.2 μm CMOS [2], and wanted to compare the performance of full-custom with FPGA technologies.

From the 4.0 ns cycle time we must subtract 0.1 ns for clock skew on the FPGA, providing 3.9 ns to travel from register to register on the chip. Using worst case timing tables for the 3195-09 device [8], it takes 1.3 ns to go from the rising edge of the clock to the CLB output, with a setup time to the CLB of either 1.5 ns for look-up table inputs or 1.0 ns for direct-in (DI) data inputs. Table II shows the overall timing budget. The X-Delay timing analyzer reports a maximum clock skew of 0.1 ns between internal CLB's for the 3100-09. A design frequency of 250 MHz allows for 1.1 ns of allowable wiring delay for normal CLB inputs and 1.6 ns of wiring delay for DI inputs. The DI inputs do not pass through the lookup tables before going to the flip-flop registers.

These wiring delays on the "-09" series allow nearest neighbor communication using the direct data lines between

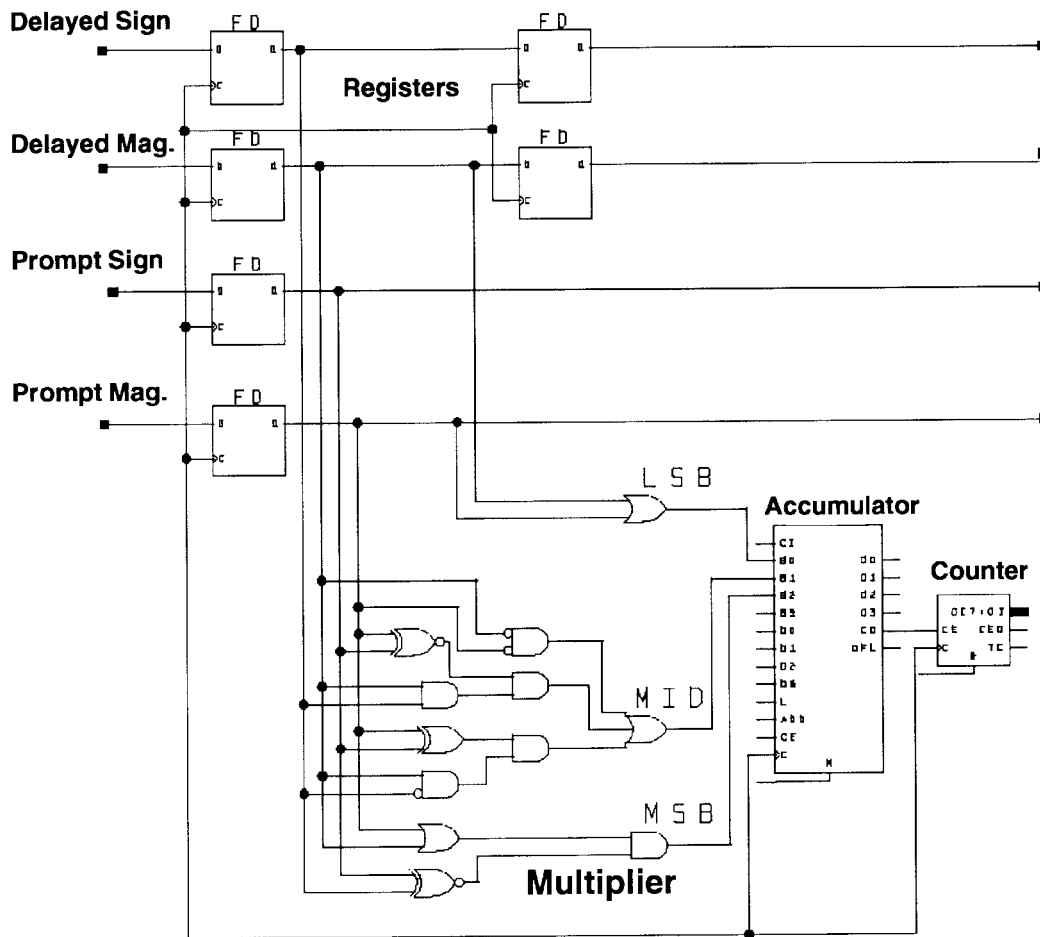


Fig. 2. Basic schematic of a single correlator lag.

CLB's within 1.1 ns, and occasionally diagonal data propagation to normal data inputs. Figs. 3 and 4 show the wiring delay from a central CLB to other nearby CLB's. Direct interconnect takes 0.3 ns, and general routing resources take significantly longer. A 250-MHz system can use any routing channel of less than 1.1 ns for general inputs and any routing channel of less than 1.6 ns for direct data inputs (DI) to the CLB registers. These constraints determine the event horizon for a 250 MHz system

Utilizing this design methodology, the entire chip can operate with worst-case cycle delays of under 4.0 ns. Future FPGA design tools could show for any selected CLB the event horizon for that CLB at a specified clock frequency, perhaps with the entire neighborhood highlighted in green. This could vividly show designers how far they could go with a signal before resynchronization.

Note that signals travel faster to the east than to the west, and slightly faster south than north. The standard orientation of the Xilinx device in the EditLCA editor defines east and west, north and south, i.e., pin 1 located in the northwest corner, facing the surface of the die. These directional biases in the 3100 family constrain the orientation of the chip relative to the desired data flow direction. When floorplanning a large chip, designers must often rotate and reflect a subcircuit for optimal packing density. Directional biases reduce the maximal packing density and operating speed of FPGA's, and

may stem from a habit of drawing schematics from left to right, and translating these schematics into chip layouts fairly literally. Newer FPGA's such as the XC4000 propagate signals at the same speed to the east and west, but the larger size of the XC4000 CLB reduces pipeline performance relative to the XC3100 in the same chip technology. The XC4000 family lost all direct interconnects, significantly increasing the time for nearest neighbor communications. The XC4000EX/XL regained direct interconnects, but only in the east and south directions, exacerbating the directionality problems of the original XC3100A architecture. Symmetry makes rotations and reflections of subcircuits possible, an important operation for building larger systems. Hopefully, future FPGA architectural designers will see the benefits of directional symmetry for routing networks in FPGA's.

#### A. Estimates for the XC4000XL Device Architecture

Recently, Xilinx has focused its high-speed efforts on the XC4000XL family. Worst case timings for this family provide a performance comparison with the 3100A family. The XC4000XL CLB has more symmetry and has internal RAM capability, useful for high-density accumulators. Early estimates indicate that a correlator lag may require 20% fewer CLB's in the XC4000XL than in the XC3100A design.

Xilinx produces the XC4000XL-1, the fastest speed grade currently available. For this speed grade, the timing budget

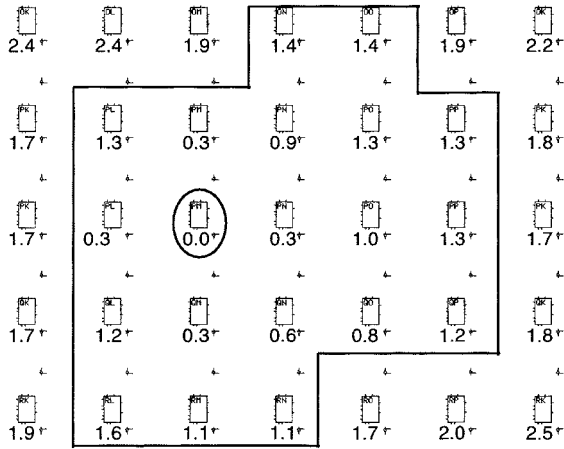


Fig. 3. The maximum distance a signal can travel in a single cycle using the DI input pin on the CLB. All data communications must lie within this event horizon to meet the timing specifications.

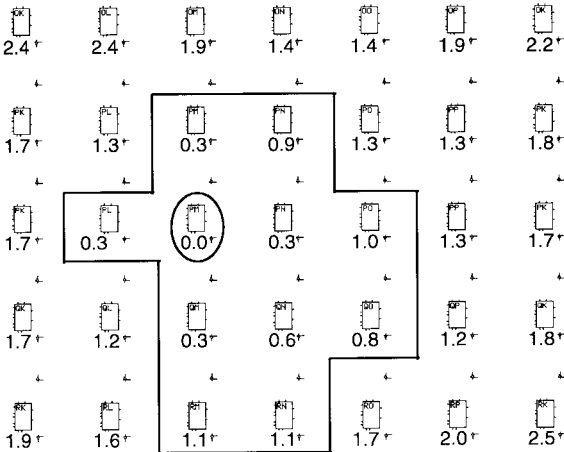


Fig. 4. The maximum distance a signal can travel in a cycle passing through a single look-up table (LUT) before synchronization. The event horizon indicates the wire limits for a desired clock frequency.

includes 0.1 ns clock skew, 1.6 ns clock to out for the CLB, 1.2 ns of routing delay, and 0.9 ns to pass through the lookup table and setup the next register. This gives a total of 3.8 ns for the CLB delays. Assuming that the clock and I/O systems can keep up, worst-case CLB timing indicates operation at 263 MHz. Xilinx has indicated that the “-09” speed grade available early in 1998 will increase the top speed of this family by 15%, suggesting maximum operating speeds of over 300 MHz in 1998. This speedup illustrates one of the important advantages of FPGA designs over ASIC designs: without any additional engineering work, the design gets faster as new FPGA speed grades become available.

IV. ON-CHIP CORRELATOR ARCHITECTURE

The schematic of Fig. 2 would probably not operate at 250 MHz by itself. The methodology described in the previous section can transform this schematic into a highly pipelined design operating at 250 MHz. For a synchronous 250 MHz system, the global clock must register every CLB output. The inputs for any look-up tables must come from nearby CLB's.

Spanning distances of 3 CLB's requires the DI pin. Each CLB has only one DI pin, which limits the maximum density of the design at times. The following Section IV-A shows how to transform the schematic of Fig. 2 into a retimed architecture meeting all of these design methodology constraints.

A. Retiming of the Lag Architecture

The lag architecture of Fig. 2 has a number of long propagation paths that benefit from pipelining for maximum-speed operation. Pipelining for a 250 MHz system requires breaking down each combinational circuit into four-input look-up tables (LUT's) immediately followed by a register. Any combinational blocks larger than four inputs must divide into two or more blocks with a new register in the middle so as to meet the single LUT rule between registers. Fortunately, some of the logic in the multiplier can combine with some logic in the 4 bit accumulator to shrink the total number of CLB's. For example, the LSB of the multiplier can combine with the half adder to produce a running sum and carry bit for the LSB of the accumulator in a single CLB. Both sum and carry pass through a register. The carry propagates during the following cycle to the next bit of the accumulator. In this way the pipelined architecture processes one bit of carry per cycle in a pipelined fashion and need to only traverse one lookup table between registers.

Two four-input LUT's in a single CLB can absorb all of the logic in the middle and MSB bits of the product. The sign and magnitude bits of the prompt and delayed data all affect these product bits. If all four input registers lie adjacent to a single CLB, then that single CLB can compute two bits of product in each cycle. Registers latch these two product bits and pass them to the accumulator for further processing.

A bit-pipelined accumulator provides maximum performance for carry chains. By breaking the carry chain into single-bit stages, carries only traverse a single LUT between each register. Additional registers delay each input operands as needed to align the inputs. The LSB accumulator computes one result each cycle and passes it to the MID accumulator on the next cycle, which passes its result to the MSB accumulator on the following cycle. The retimed schematic appears in Fig. 5, including all of the data registers required to properly retime the schematic.

B. Placement of the Retimed Schematic on the FPGA

Circuit placement on the FPGA started from the middle of the circuit out using the Xilinx EditLCA design editor (XACT version 6.0.1). As a general rule, designers should start with the most constrained part of a design and work out from there. The MID and MSB product bits have the largest placement constraints. Those bits require four inputs and generate two outputs. The four inputs must lie close to the CLB computing the outputs, suggesting a cross pattern with the MID/MSB CLB in the middle, with input data bits coming from north, south, east and west neighbors. This structure appears in Fig. 6, with the four data input CLB's forming the cross and the computation CLB in the middle. Note that for these ultrahigh-speed designs, properly positioning the data

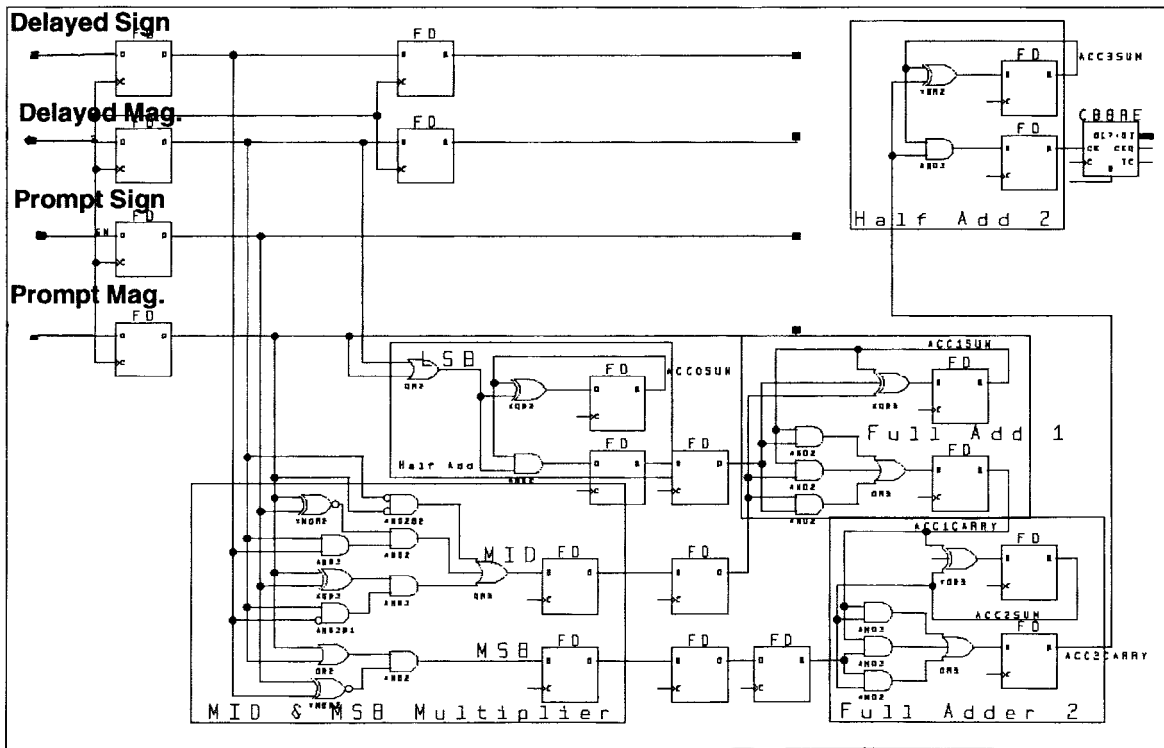


Fig. 5. Retimed correlator lag circuit that has only one four-input look-up table between registers. Each box with a large label represents a single CLB. Note the high packing density for the MID and MSB bits of the multiplier. They fit in a single CLB because only four inputs feed that circuit.

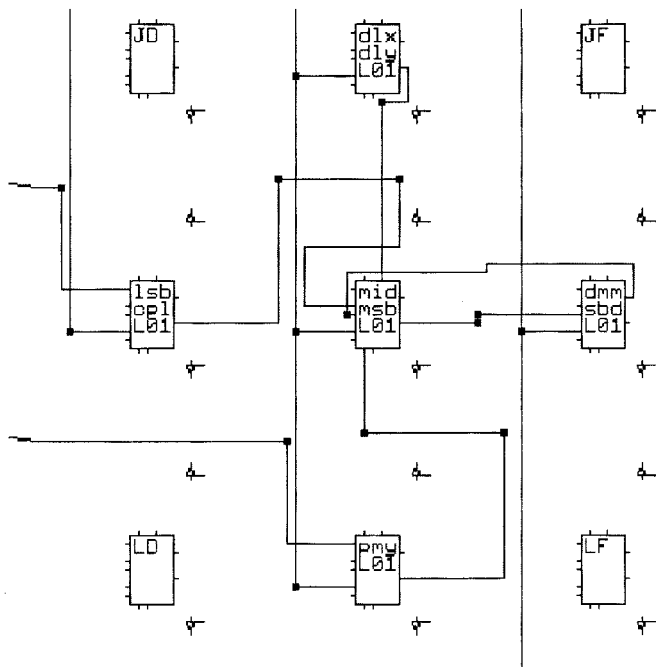


Fig. 6. CLB layout on the FPGA showing cross topology of data inputs with MID and MSB multiplier bits in the center. The four data inputs surround the multiplier CLB and provide it with input data.

takes more CLB's than performing the computation. This observation suggests that wiring speed has more importance than LUT speed for ultrahigh-performance designs. It also suggests that FPGA architectures with enhanced direct interconnect lines would improve high-performance computational density.

Since a signal can jump at most three CLB's at 250 MHz using the DI input, any lag pitch greater than three requires multiple registers for the prompt data. This observation suggests a pitch of three CLB's to maximize the wires and CLB's available while maintaining a single-hop distance between lags as shown in Fig. 7. Placing the LSB logic to the northwest of the MID/MSB product permits nearest-neighbor communication to all the product CLB's, since the four data inputs can lie in any permutation around the cross. The wiring delay from the LSB product to the MID accumulator bit requires two pipeline stages. Two pipeline stages permit the LSB result to pass south by two rows, and the other input data has to pass through an equal number of registers for synchronization with the LSB accumulator results. Wiring delay can often limit performance more than CLB delay.

### V. BOARD ARCHITECTURE

The reconfigurable correlator board can process input signals at 250 MHz and can demonstrate chip-to-chip communications at that speed as well. Fig. 8 shows the layout of the board. Positive ECL (PECL) signals pass through 50-Ω transmission lines through SMA connectors (small, coaxial 50-Ω matched impedance connectors) onto the left and bottom sides of the board. The lines drive PECL-to-TTL translators placed close to the FPGA's with minimum loading of the TTL lines. Short TTL lines one centimeter in length propagate very high-speed TTL signals at 250 MHz with minimum loading. The TTL signal passes onto the FPGA and into registers for the input data. The clock signal runs through the same PECL-TTL converter and onto the global clock pin on the FPGA.

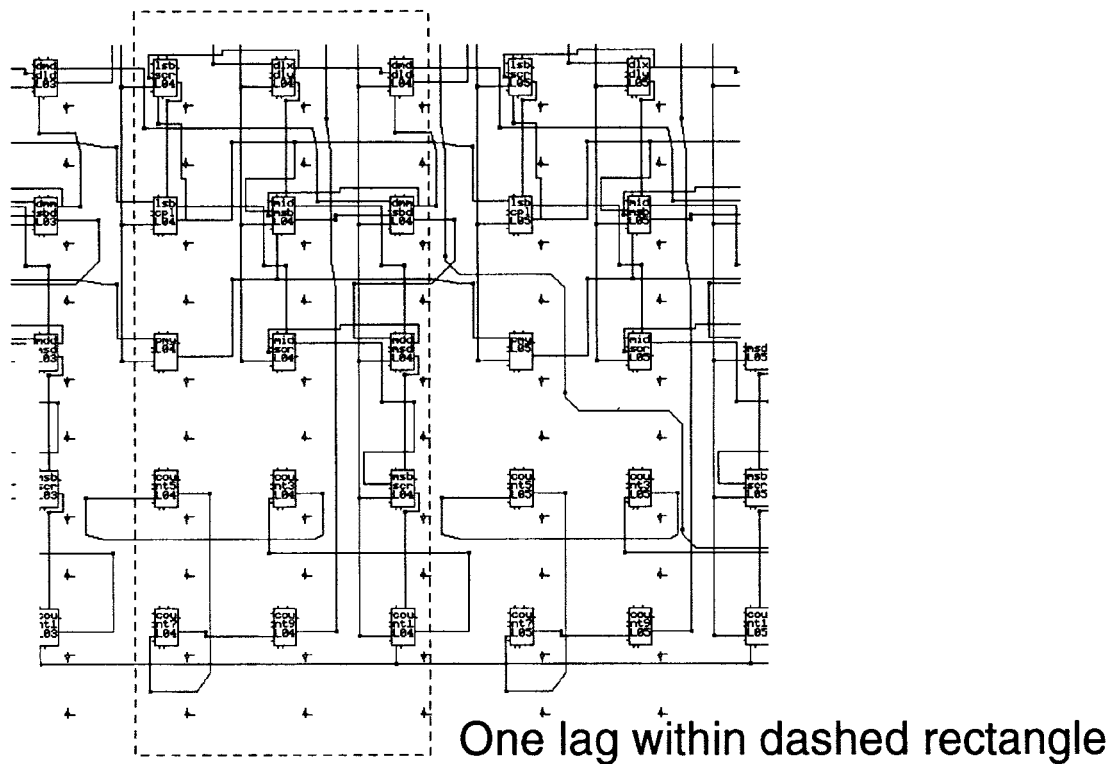


Fig. 7. Layout of a correlator lag on the FPGA. These lags tile adjacent to one another to form a two-dimensional array of lags across the correlator chip. The accumulator lengths adjust to trade off the integration time for packing density of the lags.

The surface traces between the two FPGA chips (not shown in Fig. 8 but near C30 and C31) also must run at high speed. These one-centimeter traces do not have any vias, minimizing board capacitance. An estimated capacitance of 5 pF for the PQFP-160 pin and an estimated trace capacitance of <5 pF results in a total load delay of less than 10 pF. The Xilinx data sheet for the 3100A-09 indicates a clock to output time of 1.45 ns for an unloaded pin and 3.3 ns into 50 pF for the high-speed output pad, and a worst case set up time of 8.1 ns, for a total of around 10 ns into 10 pF. Typically the XC3100A performs much faster. The setup time specifications for the Xilinx 3100A series have always been problematic. In practice, we transmitted data at speeds of well over 250 MHz on the engineering samples we received without any problems. If we had encountered data loss at this speed, we would have demultiplexed the data two or three times and run at 125 or 83 MHz, slow enough to meet the worst-case timing spec. Since we had only four data pins to transmit, demultiplexing to 8 or 12 pins would have been practical if required.

## VI. EXPERIMENTAL RESULTS

We tested the correlator circuit using an HP 80000 data generator and observed correct operation at speeds of over 250 MHz on the bench under ambient conditions with no forced-air cooling. Table III shows an example of deterministic pattern testing at 250 MHz. A regular pattern of input samples pass to the prompt and delayed data inputs of the correlator chip from the HP 80000, and the Xilinx readback circuit retrieves the counts in the correlators. In this way the HP80000 can test all of the products in the multiplication table, and confirm correct

operation of the integration circuit. For 8 388 608 counts, we obtained exact results of 0x600 000 and 0 for the products of 11 and 01, confirming the four corners of multiplication Table I. Further testing provided perfect experimental results for the other products at 250 MHz. In particular, the test illustrated in Table III exercises all the products of the multiplier. Lag 0 combines the products  $(6, 3, 3, 6) = 4.5$  with equal weights. Lag 1 averages products  $(6, 4, 3, 4) = 4.25$ . Lag 2 combines  $(2, 2, 4, 4) = 3.0$ . Lag 3 combines  $(0, 2, 3, 2) = 1.75$ , and Lag 4 combines  $(0, 3, 3, 0) = 1.5$ . These experimental results correspond exactly to their hexadecimal equivalents measured over 8 million integration periods as shown in the chart. Lags 4–7 mirror lags 3–0, and lag 8 equals the results in lag 0. These results confirm correct operation of the correlator under periodic conditions at 250 MHz.

After the deterministic tests, we programmed the HP80000 to generate a pseudo-random signal for the correlator. The zero lag should have a positive correlation for a random signal, while the remaining lags should have no correlation out to the length of the pseudo-random sequence. The correlation data passed into a PC using the readback feature of the Xilinx XC3100A series, which permitted the analysis of the data and comparison with expected results for different clock frequencies.

In actual operation of the XC3100A correlator, the readback feature can take a “snapshot” of all the accumulation registers in the chip. The integration halts for a period from 15 to 20 ns while a readback trigger occurs, which copies the contents of all the accumulation registers into the readback shadow registers on the FPGA. Integration resumes after the 15-ns

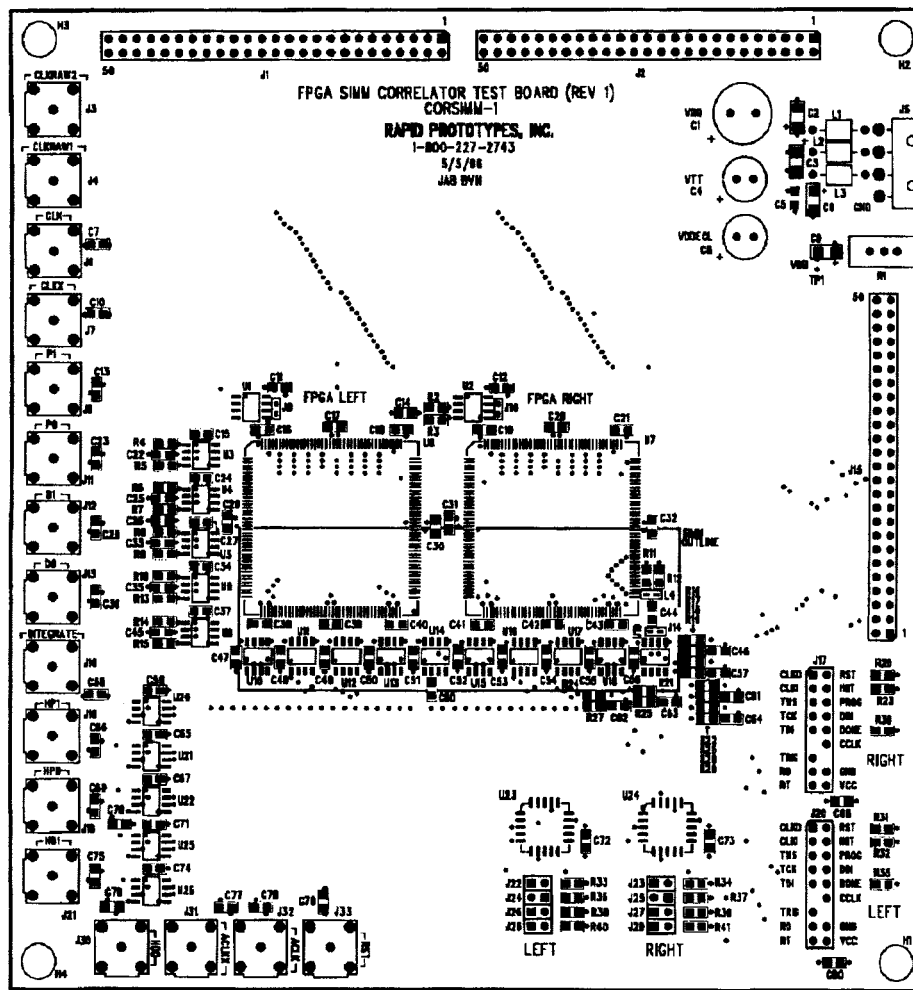


Fig. 8. Reconfigurable correlator board layout. Inputs pass through the left and bottom SMA connectors. The FPGA chips lie in the center, and the PECL to TTL translator chips surround the FPGA chips. Note that the translators and the FPGA's lie within roughly 1 cm of each other to minimize capacitance.

TABLE III

PERIODIC PATTERN RESULTS AT 250 MHz FOR A PATTERN WITH A PERIOD OF EIGHT CYCLES. (11) = -3, (10) = -1, (00) = +1, (01) = +3

Prompt	Delayed	# of Samples:	Correlator Counts
		Lag	
11	11	0	0x800000
11	11	1	0x480000
10	10	1	0x440000
00	00	2	0x300000
01	01	3	0x1c0000
01	01	4	0x180000
00	00	5	0x1c0000
10	10	6	0x300000
11	11	7	0x440000
11	11	8	0x480000
10	10	9	0x440000
00	00	10	0x300000
01	01	11	0x1c0000
01	01	12	0x180000

TABLE IV

CORRELATOR COUNTS SHOWING PERFORMANCE AT SPEEDS FROM 10 MHz UP TO 250 MHz. THESE TESTS INTEGRATED 2<sup>24</sup> SAMPLES

Channel	10 MHz	100 MHz	200 MHz	250 MHz
Lag 0	0x600000	0x600000	0x600000	0x600000
Lag 1	0x600002	0x600002	0x600002	0x600002
Lag 2	0x600001	0x600001	0x600001	0x600001
Lag 3	0x600003	0x600003	0x600003	0x600003
Lag 4	0x600003	0x600003	0x600003	0x600003
Lag 5	0x600002	0x600002	0x600002	0x600002
Lag 6	0x600002	0x600002	0x600002	0x600002
# Counts	0xFFFFFFFF	0xFFFFFFFF	0xFFFFFFFF	0xFFFFFFFF

settling time of the readback trigger. The readback bits pass at leisure to the PC while integration continues on the device. In this we can make use of the double-buffering function of the normally invisible readback registers.

We verified the correlator from dc to 250 MHz using this readback method. Table IV shows the performance of the cross

correlator at speeds from 10 to 250 MHz. Residuals came from start-up conditions and internal pipelining on the correlator chip. The results agreed for all frequencies including 250 MHz.

We performed a simple power measurement for the correlator using pseudorandom input waveforms at 250 MHz. The entire board consumed 0.90 A using 7, mostly due to the ECL circuitry. Removing four of the lags resulted in a current consumption of approximately 0.80 amps. Therefore each lag draws roughly 25 mA of current during operation. A 3.3-V

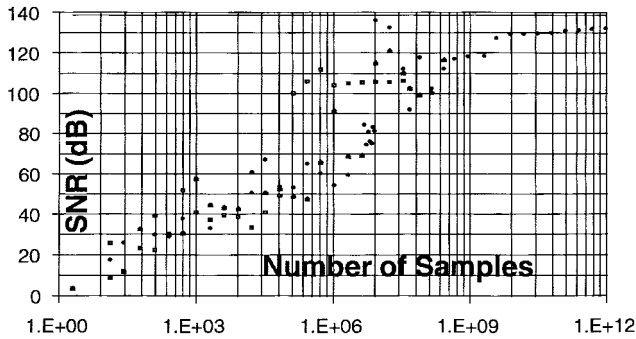


Fig. 9. Integration performance of the FPGA cross correlator running at 250 MHz using pseudorandom inputs.

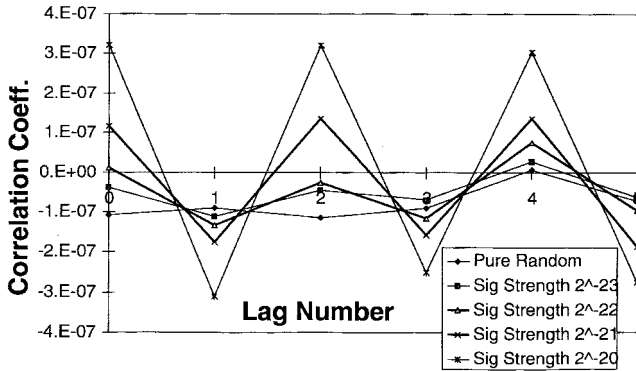


Fig. 10. Correlation of a pseudorandom stream of  $2^{27}$  uncorrelated samples with a weak Nyquist signal added.

FPGA would use half as much power as a 5-V FPGA. Newer generation parts will consume less power because of their smaller features sizes compared to current chip technologies.

Fig. 9 shows the long-term integration results for the FPGA correlator. The line at the top of the chart at 138 dB represents the theoretical limit to performance for a  $2^{23}$ -sample pseudorandom sequence. The chart shows how the correlator integrates asymptotically to the theoretical limits. We obtain a signal to noise ratio of over 130 dB for integrations of  $10^{12}$  samples.

The experimental frequency limit stems from the TTL buffer driving the global clock circuit. We performed an experiment using the direct CMOS clock input pin, and got ripple counters to work at speeds of over 600 MHz on the engineering samples obtained of the -09 speed grade. This result suggests that FPGA's could operate faster than 250 MHz if they used self-timed signaling instead of global synchronous signaling, since clock distribution limits the speed of the system. This approach could also reduce power supply bypassing since fewer inputs would switch simultaneously on the clock edges. Future experiments may validate this conjecture.

Fig. 10 shows the correlator performance for very weak signals masked by Gaussian noise. The horizontal line at zero represents the cross correlation of a purely random signal of  $2^{23}$  samples. An introduced Nyquist signal causes alternating positive and negative correlations in adjacent lags of the correlator. Statistical edge effects of the pseudorandom word stream caused a small residual correlation of the random  $2^{23}$

samples. Weak signal correlations overshadow the edge effects even at strengths of  $2^{-21}$  of the energy in the random signal.

VII. FLEXIBLE CORRELATOR PRECISIONS

FPGA's can reallocate the computations in a correlator array. Depending on the object under observation, the ideal correlator has different specifications. For example, when observing a stellar formation region such as the Orion Nebula [9], astronomers want many channels of resolution to measure precise Doppler shifts and radial velocities of the molecular gas clouds surrounding the proto-stars under formation. In this case, a 1-bit correlator with low sensitivity but many channels provides peak performance. Conversely, weak signals from distant galaxies require great sensitivity with fewer channels [10]. In these cases, a 2-bit or even 3-bit correlator produces the best results, but with fewer correlator lags.

In a conventional signal processing array, the worst case of many correlator lags at the highest precision might consume as much silicon area as an equivalent FPGA system. A reconfigurable array can reallocate its resources to provide more lags at low precision or fewer lags at higher precision. This flexibility can make up for the area overhead incurred by the FPGA support logic. If new algorithms arise, the reconfigurable processor can accommodate them, while the fixed hardware array becomes obsolete. Therefore, a reconfigurable array has greater longevity and usefulness than a fixed gate array. Finally, as IC processes improve, so does the performance of the reconfigurable array. The same reconfigurable design can run on faster FPGA chips without any redesign of the correlator circuit, and without incurring additional engineering costs to produce new chips. A fixed gate array shrunk to a new process still requires new tooling and masks, thereby incurring significant costs to make the faster chips. All these factors contribute to the attractiveness of performing real-time computations on reconfigurable FPGA arrays.

VIII. EXTENSIONS TO PIPELINED SIGNAL PROCESSING

The basic technique illustrated in this paper can generalize to other real-time signal processing applications. The tasks include outlining a pipelined architecture, decomposing the signal processing into small blocks that fit into a single CLB with local communications and synchronous registers used for every CLB output, placing the CLB elements starting with the most constrained interconnections and working outward to the rest of the design. By specifying ahead of time the required speed of the system, and by constraining oneself to wiring paths that fall within the timing constraints, designers can build large circuits that meet timing requirements *a priori*. This approach differs from the technique of automatically placing and routing all the signals in the chip and hoping they meet the timing constraints. Achieving timing goals by construction makes the the implementation process predictable, resulting in much more deterministic performance for high-speed systems.

Feed-forward systems with high throughput requirements and low sensitivity to latency work best with this design methodology. Systolic arrays fall into this class of computation. Feedback systems can also benefit from these pipelining

approaches, but the feedback loops may require retiming. Circuits with multistage pipelining often achieve minimum delay-area products, even though they use more registers than a minimum-area approach. The best FPGA's for this design approach have many registers and local fast wiring resources. These two factors provide the highest performance in real-time FPGA computations.

How could current FPGA architectures improve for these types of applications? First of all, the direct interconnect lines could double in number to include XQ and YQ outputs going east, west, north and south. This would double the potential throughput for nearest-neighbor communications. Second, avoiding switch matrices out to a distance of two or three CLB's could speed up the signal flows at these distances. Every switch adds significant resistance to the routing channel. Third, since the clock signal propagates through the distribution network slower than the data signals, wave pipelining the clock through numerous buffers would permit clock propagation at higher frequencies. Asynchronous techniques might also work well. Fourth, a second DI pin on each CLB would permit maximal use of the two registers in each CLB. Finally, a symmetric FPGA architecture in all directions helps system design immensely. Without symmetry, designers cannot rotate and reflect subcircuits as needed for minimum area. Closure under reflection and rotation would improve reusability of subcircuits under maximum packing conditions, and improve overall FPGA performance.

## IX. CONCLUSION

This work has demonstrated that FPGA's can successfully perform real-time signal processing at 250 MHz. We have verified correct operation of a 2-bit cross correlator using a variety of periodic and pseudo-random integrations of up to  $10^{12}$  samples, and have validated translation circuits from PECL to TTL levels outside the FPGA. We have also demonstrated chip-to-chip transfers between FPGA's at 250 MHz using the same techniques. These results confirm that paying careful attention to layout and pipelining can result in very high performance from FPGA's. For a given feature size, the FPGA seems about two times slower but more flexible than a full-custom implementation. The FPGA circuit retains the flexibility of reconfiguration, creating the possibility of trading off correlator sensitivity for spectral resolution.

These advantages combine to make FPGA's attractive for implementing novel signal processing circuits that operate at hundreds of megahertz.

## ACKNOWLEDGMENT

The author would like to acknowledge J. Brunetti who completed the high-speed board design and board layout for the 250 MHz FPGA correlator test fixture.

## REFERENCES

- [1] K. L. Pocek and J. Arnold, in *Proc. IEEE Symp. FPGA's Custom Comput. Machines (FCCM '97)*, Napa Valley, CA, Apr. 11-18, 1997.
- [2] B. Von Herzen, "VLSI partitioning of a 2-Gs/s digital spectrometer," *IEEE J. Solid-State Circuits*, vol. 26, pp. 768-772, May 1991.
- [3] B. F. C. Cooper, "Auto-correlation spectrometers," in *Methods of Experimental Physics, Vol. 12, Part B: Astrophysics and Radio Telescopes*. New York: Academic, 1976, p. 280.
- [4] T. G. Phillips and D. B. Rutledge, "Super-conducting tunnel detectors in radio astronomy," *Scientific Amer.*, May 1986, p. 78.
- [5] D. Woody, D. Vail, and W. Schaal, "Design, construction and performance of the Leighton 10.4 meter-diameter radio telescopes," in *Proc. IEEE Design Instrumentation of Antennas for Deep Space Telecommun. Radio Astronomy*, vol. 82, 1994, p. 673.
- [6] The CSO-JCMT Interferometer, [Online]. Available FTP: <ftp://lapakahi.submm.caltech.edu/doc/inter.html>.
- [7] *The Programmable Logic Data Book*, 3rd. ed. San Jose, CA: Xilinx, Inc., 1994 (revised Apr. 1995).
- [8] Xilinx XC3100A FPGA Family Product Specifications, Xilinx, Inc., San Jose, CA, Version 4.1, Oct. 1995.
- [9] E. Falgarone and T. G. Phillips, "Small-scale density and velocity structure of a molecular cloud edge," *Astrophys. J.*, 1996.
- [10] C. E. Walker, F. N. Bash, and R. N. Martin, "The starburst properties of M82, M83 and IC 342," *Rev. Mex. Astro.*, vol. 27, p. 203, 1994.



**Brian Von Herzen** was born in La Jolla, CA, in 1959. He received the B.S. degree from Princeton University, Princeton, NJ, in physics and studied VLSI and computer graphics at California Institute of Technology, Pasadena, where he received the Ph.D. degree in computer science in 1988.

After graduating from California Institute of Technology, he led a project at the Caltech Submillimeter Observatory and developed high-performance custom correlator chips. In 1991, he joined Synaptics, Inc., as an applications research manager and developed commercial applications of neural networks using FPGA technology. In 1993, he founded Rapid Prototypes, Inc., Carson City, NV, which focuses on reducing the development time for new commercial products using programmable logic devices. His interests include reconfigurable hardware and real-time video processing using FPGA's.