White Paper

FPGA Coprocessing Evolution: Sustained Performance Approaches Peak Performance

FPGA architecture allows for many algorithm implementations where the sustained performance is much closer to the device's peak performance when compared to quad-core CPUs or GPGPUs. The strong benchmarking results from FPGA accelerators will continue to improve with appropriate focus on the silicon, arithmetic, and library foundations. As even the largest FPGAs currently consume less than 30W of power, FPGA roadmaps are well below datacenter power and cooling limitations.

Introduction

Some key trends are converging to make FPGA acceleration of algorithms more attractive, including:

- *FPGAs large enough to encompass larger algorithms:* It is now possible to fit options-pricing algorithms or 1M-point fast Fourier transforms (FFTs) into FPGAs. The latency to offload an algorithm from the CPU to an FPGA is minor compared to the time saved by the FPGA's algorithmic speedup. For instance, it can take as little as ~1 µs in round-trip latency for the FPGA to process and return a result. As one example, when using an FPGA for Monte Carlo simulations for Black-Scholes algorithms with 1M paths per option, the computation time is shortened by more than 3,600 ms when compared to a quad-core CPU.
- *Single-core CPUs have hit a power and cooling wall:* While the silicon-process geometries produce faster single-core CPUs, they consume too much power and cooling. The move to multi-core CPUs is well underway, with AMD and Intel now shipping quad-core CPUs. However, existing software written for single-core CPUs must be rewritten to extract parallelism for reasonable performance scaling to CPUs with more cores.
- *AMD and Intel actively support FPGA coprocessing:* In some cases, these CPU interfaces—AMD with the Torrenza initiative, and Intel by licensing FSB and QPI to FPGA vendors—support 8 GB/s and latency for posted writes of under 140 ns.

Benchmark results show what FPGAs are capable of. XtremeData (XDI) has benchmarked Monte Carlo results of 1.8G results per second in double-precision floating-point logic for the XD2000i (the XDI Xeon socket accelerator shown in Figure 1) compared to 240M results per second for two quad-core CPUs, or 900M results per second in single precision for general-purpose GPU (GPGPU) (benchmarked by RapidMind*(1)*). While the FPGA's absolute performance advantage (Table 1) is significant, the performance per watt advantage is equally, if not more, impressive.

Figure 1. XtremeData XD2000i Coprocessor for Intel FSB Socket

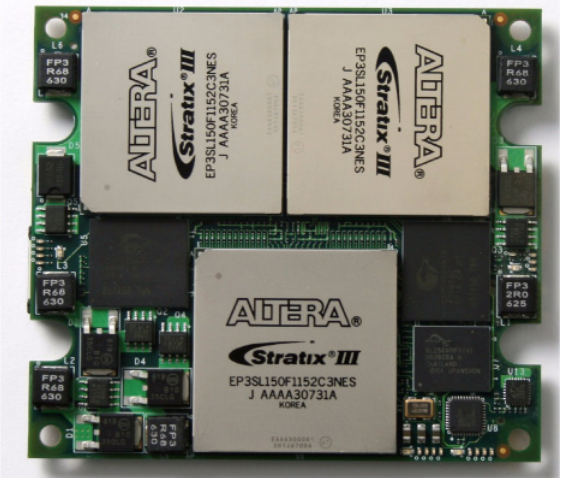


Table 1. Monte Carlo Black-Scholes Performance Comparison

Monte Carlo	2 Quad-Core CPUs	nVidia 8800	XD2000i + 2 EP3S260s
Precision	Single	Single	Double
Paths per second	~240M	~900M	1.8G
RNG type	Halton	Halton	Mersenne
Source	RapidMind	RapidMind	XtremeData

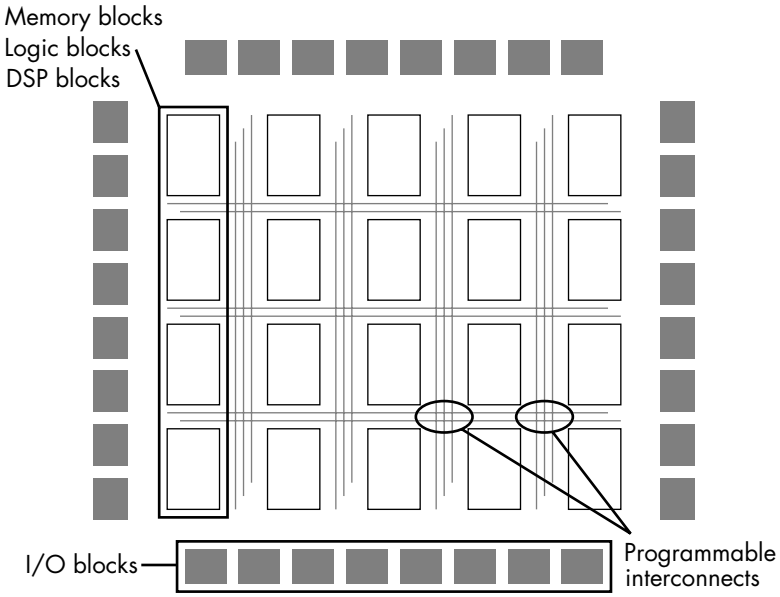
Note:
CPU and GPU results at 1M paths per option.

How does an FPGA running at 150 to 250 MHz deliver better results than a quad-core CPU running at 2 to 3 GHz or a 128-core GPGPU running at 1.35 GHz? In the case of the Monte Carlo Black-Scholes algorithm, the FPGA architecture has unique capabilities (parallelization, custom instruction with pipelining, large primary cache, and flexible datapath routing) that combine to deliver those benchmark results.

FPGA Architecture Features

FPGAs are flexible devices that can be programmed and reprogrammed as needed. As shown in Figure 2, a typical FPGA is made up of an array of logic blocks, memory blocks, and digital signal processing (DSP) blocks, all of which are surrounded by programmable interconnects that can be configured with software.

Figure 2. FPGA Architecture



This architecture enables the following features:

- *Function parallelization:* replication of a function multiple times
- *Data parallelization:* handling of data arrays or data matrices
- *Custom instruction with pipelining:* streaming data with one result per clock
- *Huge primary cache bandwidth and size:* 3X to 10X compared to GPGPU
- *Flexible routing of datapath:* huge crossbar routing transfers data in one clock
- *Concatenation of functions and data flow:* all in one clock
- *Custom off-chip I/Os:* protocol, bandwidth, and latency as needed
- *Scalable roadmap:* larger arrays have plenty of headroom for power and cooling

FPGA advantages for parallelization and pipelining are well understood. FPGAs also have an advantage in primary cache and bandwidth over GPGPUs. Within the FPGA, logic is surrounded by memory blocks. Using the Altera® Stratix® III FPGA EP3S260 chip as an example (Table 2), there are 48 large (M144K) blocks, 864 medium (M9K) blocks, and 5100 small blocks (MLABs), all with memory protection and parity protection. The M144K blocks have error correction coding (ECC) built in, and programmable logic can be used for ECC on the other two types of memory. While MLABs are useful for constants or intermediate results, the M9K and M144K blocks show that the XDI XD2000i module with two EP3S260 chips has a 3.3-Mbyte primary cache with bandwidth of 3.8 Terabytes/sec. This is about five to ten times larger than the primary cache (for streaming processors) on the nVidia 8800 GTX GPGPU.

Table 2. Memory Resources on the Stratix III FPGA

Per EP3S260	Quantity	Kbytes	Total Kbytes	Bytes Width	# of Ports	Clock (MHz)	Gbytes/sec
MLABs	5100	0.032	163	2	2	250	5100
M9K blocks	864	1	864	4	2	250	1728
M144K blocks	48	16	768	8	2	250	192
Total			1795				7020

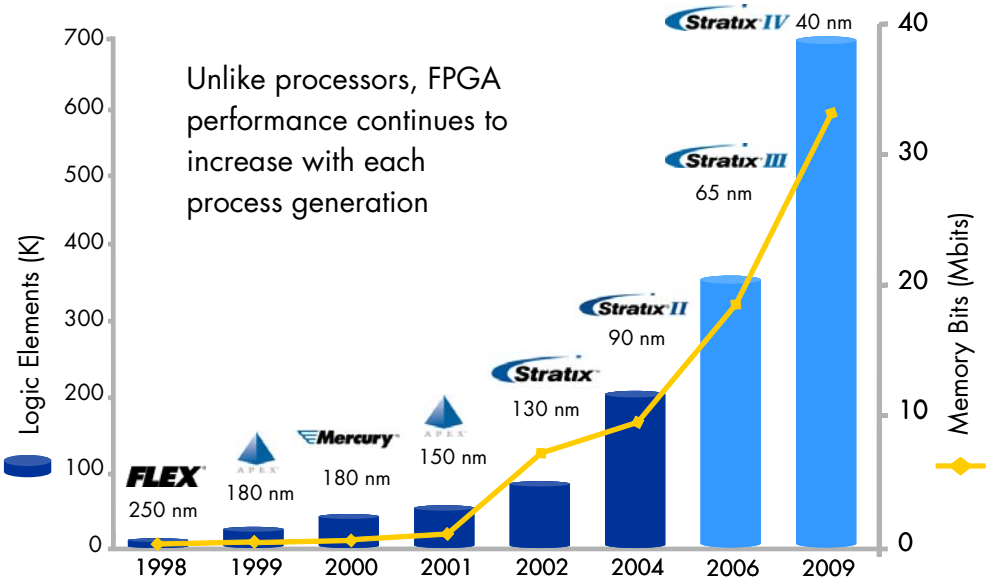
The advantage is more than just a higher primary cache size and greater bandwidth—it is the FPGA's ability to use the excess routing bandwidth to construct the data path and memory access flexibly and directly to each logic block. The programmable interconnects shown in Figure 2 provide a large amount of routing bandwidth.

Modules and boards can be designed as needed for off-FPGA bandwidth, memory size, and latency. There is also the ability to customize the I/Os around the FPGA, such as:

- XDI's Opteron "in socket" XD2000F accesses the motherboard/blade DRAM
- SRC's H-MAP accesses 19.2 Gbytes/s (8 banks) of SRAM
- Bypassing the operating system and taking Ethernet traffic straight into the FPGA for lower latency

A final and important aspect of FPGA architecture is the ability to scale to larger arrays of logic, memory blocks, and DSP blocks. Figure 3 shows how logic and primary cache sizes scale together. With the largest FPGA devices currently consuming about 30W at peak performance, there is plenty of room for FPGA architecture to scale to new process geometries without hitting current datacenter power and cooling limits.

Figure 3. FPGA Density



Building the Foundation

While FPGA architecture has some nice capabilities, several areas must come together for a good CPU coprocessing solution:

- *Silicon foundation:* Logic, DSP, and power management
- *Arithmetic foundation:* Operator cores optimized for FPGAs and datapath compiler
- *Library foundation:* Function optimized for specific FPGA resources
- *System level:* Coprocessing tool chain with CPU interface bandwidth and latency

Performance for a class of algorithms can be increased significantly by focusing on the three foundation areas, as shown in the following examples of recent improvements in single- and double-precision floating point.

Silicon Foundation

Most algorithms using double-precision floating point have an approximately one-to-one ratio of addition and multiplication operators. While the addition core uses FPGA logic, the multiplication core uses the DSP blocks, so the FPGA must have a balanced ratio of logic to DSP blocks. The DSP blocks also must be able to minimize the number of blocks required to handle 54-bit-by-54-bit mantissa multiplication.

See Altera's white paper, *Designing and Using FPGAs for Double-Precision Floating-Point Math (2)*, to learn about the Stratix III and Stratix IV FPGA families' ideal ratio of logic and DSP blocks with efficient DSP block design.

Another feature that helps with the performance-to-power ratio is Altera's Programmable Power Technology, which allows every logic block, DSP block, and memory block to be programmed to run in higher or lower power mode depending on the design timing requirements. With only the timing-critical blocks set to high-speed mode, power dissipation in Stratix III and Stratix IV devices is reduced substantially.

Arithmetic Foundation

Floating-point operator cores have been improved to run at higher clock speeds, to use fewer DSP blocks, and to use less logic. Altera also has developed a floating-point compiler to reduce the logic required to route 64-bit data paths between different floating-point operator cores.

Altera's white paper *Floating Point Compiler Increasing Performance With Fewer Resources (3)* describes how the combined steps of normalizing (converting fixed format to floating-point format) at the end of one floating point operation and then denormalizing (converting floating-point format to fixed format) for the input of the next floating-point operation can be significantly reduced. The entire datapath for a mathematical expression with floating-point operations can be fused together, saving up to 40 percent in logic and increasing clock speed slightly.

Although single-precision and double-precision formats have different strengths, an important point is that the FPGA can handle any mix of precision. Most accelerators use fixed single-precision or double-precision logic, but the FPGA can be configured as required for an algorithm. In fact, it is also possible to use extended single or other precisions between single and double precision with development of appropriate operator cores. For some applications, just a bit more precision is needed than the 23-bit mantissa in single precision, but it is not very "green" to use devices with double-precision logic and 52-bit mantissas when not much more than 23 bits is required.

The right mix of floating-point capabilities applies to other operators, too. If an algorithm has many transcendentals (exponents, logs, etc.), then the FPGA can be configured with as many as are needed. In GPGPU designs, a few hard blocks are added for such functions but at a much smaller ratio than single-precision floating-point logic.

Library Foundation

Function libraries are needed to employ algorithmic tricks, abstract the hardware details, and optimize for that particular FPGA's resources. For the Intel x86 architecture, the Math Kernel Library has a set of functions that optimize for a particular CPU cache size, instruction set, etc. As an example, a FPGA matrix multiplication function takes advantage of:

- *Matrix blocking:* Each matrix can be subdivided into sub-blocks
- *FPGA data reuse:* One sub-block row or column is kept in FPGA memory while the other sub-block is read in
- *Double buffering:* The next sub-block row or column is read in ahead of time

Using the above approach and the floating point compiler, an EP3S260 matrix multiplication core has been developed that allows streaming of 192 x 192 sub-blocks into the FPGA at the sustained performance rate of 47 GFLOPS compared to the device peak performance of 49 GFLOPS. Examples of Altera's libraries of functions include:

- xGEMM
  - SGBMM
  - ZGBMM
  - DGEMM
- FFT (double precision)
  - Streaming (up to 16K)
  - Block (256K to 1M pt)
  - Single-precision FFT: intellectual property (IP) core designed with floating-point compiler
- LU decomposition
- Cholesky decomposition
- Black-Scholes

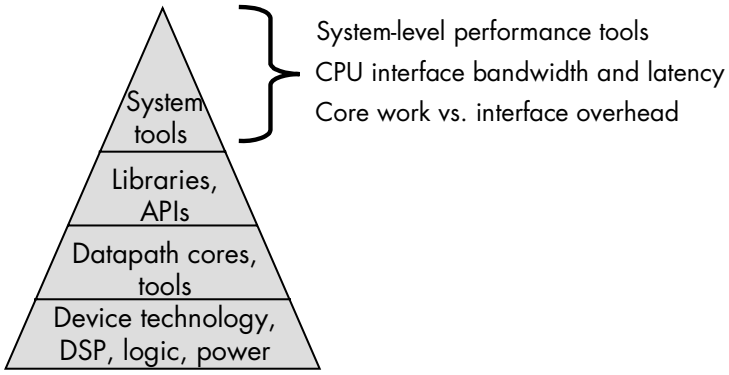
Examples of partner (SRC, XDI, Mitrionics, Impulse-C) libraries of functions include:

- Image and signal processing
  - 1D and 2D FFT
  - DCT, IDCT
  - Image filtering, image matching, image fusion, image enhancement
  - Histogram, global image analysis
  - Cross-correlation, color space transformations
  - Filtered backprojection
  - Wavelet compression
  - Sobel convolution
- Encryption
- Application/algorithm examples
  - Monte Carlo Black-Scholes
  - BLASTn, BLASTp

System Level

Building upon the silicon, arithmetic, and library foundations, the system-level solution shown in Figure 4 takes care of the tool chain, module/board design, CPU interface, and data passing from the CPU to the FPGA-based accelerator. This is the domain of partners such as AutoESL, Impulse, Mitrionics, SRC, and XDI.

Figure 4. Foundation for FPGA Acceleration



FPGA acceleration solutions include:

- *Appliances:* Systems with fixed acceleration capabilities
- *Function acceleration:* Frequently the equivalent of an x86 API call
- *"Daily algorithm" acceleration:* Allows daily changes to an accelerated algorithm

Purpose-built appliances already shipping include Activ's financial market data acceleration, SRC's image tracking for unmanned aerial vehicles, XDI's database acceleration, and bioinformatics appliances. In the function acceleration category, functions such as random number generation for the financial industry and 1M-point FFT for military are garnering attention. In the daily algorithm area, tool chains such as SRC CARTE, Mitrionics C, and XDI's MATLAB for the XD2000i are providing the flexibility needed to make changes to algorithms and to the FPGAs that accelerate them on a daily basis.

Enabling Sustained Performance Close to Peak Performance

For algorithm tasks that can be parallelized or pipelined, FPGA capabilities frequently enable a much higher sustained performance relative to peak performance and all the device resources available. Three different scenarios are described: a blade with two quad-core CPUs, a blade with one CPU and an XD2000i, and rack-mounted GPGPUs, each of which is connected to a corresponding CPU in the same rack. Using HP servers, 64 BL260c blades fit in a 42U rack, while 56 CPU/GPU pairs fit using the DL160G5 and nVidia Tesla S870. In the power calculations, the lightly loaded accelerator companion CPU is estimated at 45W versus 75W for dual-socket CPUs running a workload. (These approximated power estimates are based on recent HP Java benchmarking.*(4)*

Starting with the example of XDI's Monte Carlo Black-Scholes benchmark, a Black-Scholes equation pipeline was created to run at 150 MHz. On each clock, random numbers generated on the FPGA by a Mersenne Twister core were fed (concatenated) into this "custom instruction" with one result per clock. Twelve of these custom instruction pipelines fit in the two EP3S260 FPGAs on the XD2000i module, yielding 12 × 150M = 1.8G results per second with double-precision floating-point logic. With additional tuning, XDI expects to achieve twice this performance.

For additional information on this benchmarking, refer to XDI's white paper *FPGA Acceleration of European Options Pricing (5)*.

It is interesting to compare sustained performance in this Black-Scholes benchmark with the peak-performance floating-point capabilities of the different architectures. Looking at the architecture of a quad-core CPU running at 2.5 GHz and the nVidia 8800 GTX, Table 3 shows the peak performance for single-precision floating point, as well as the peak performance of a completely routed EP3S260 in both single- and double-precision floating point using the floating point compiler

Table 3. Peak Floating Point Performance

1:1 Adder/Multiplier	2 Quad-Core CPUs	nVidia 8800	XD2000i w/ 1 EP3S260	XD2000i w/ 1 EP3S260
Precision	Single	Single	Single	Double
Clock (MHz)	2500	1350	280	256
FP ops/cycle	8	2	1	1
# of cores	8	128	384	192
Peak GFLOPS	160	346	108	49

Note:
(1) FPGA peak performance calculated with floating point compiler.

Since the Black-Scholes equation requires more functions (exponent, square root, etc.) than the usual addition and multiplication functions, the total GFLOPS in the Black-Scholes results are not counted. Table 4 shows the ratio of Black-Scholes results to peak GFLOPS to get a relative measure of sustained performance versus peak performance.

Table 4. Monte Carlo Black-Scholes Performance/Power Ratio

Monte Carlo	2 Quad-Core CPUs	CPU + nVidia 8800	CPU + XD2000i w/ 2 EP3S260s
Precision	Single	Single	Double
M results per second	240	900	1800
Performance/peak GFLOPS	1.5	2.6	12.0
Power	~150W	~215W	~110W
Performance/watt	1.6	4.2	16.4
Results/clock	0.1	0.7	6.4

The FPGA has the best sustained performance compared to peak performance. The FPGA also has the best raw performance in double-precision logic compared to single-precision logic for the other two accelerators, and the best performance per watt.

As another example, for matrix multiplication of 1024 x 1024 matrices, the recent HP benchmarking *(6)* measured sustained performance for a quad-core CPU at 2.66 GHz and for a GPGPU. Altera has a matrix multiplication core that sustains 96 GFLOPS single-precision logic and 47 GFLOPS double-precision logic by using a double-buffering scheme where the next set of input matrices is brought into FPGA memory ahead of time. A 5 percent overhead allocation for communications between the FPGA accelerator and the CPU provides the estimates in Table 5.

Table 5. Matrix Multiplication Sustained Performance vs. Peak Performance

Matrix Multiplier	1 Quad-Core CPU	nVidia Tesla S870	XD2000i w/ 1 EP3S260	XD2000i w/ 1 EP3S260
Precision	Single	Single	Single	Double
Sustained GFLOPS	70	70	97 (estimated)	44 (estimated)
Perf./peak FLOPS	~85%	20%	90%	90%

Note that the CPU efficiency is good relative to the floating-point logic, and that raw performance is similar to the FPGA module. Table 6 shows the performance per watt of a CPU paired with an accelerator or another CPU, with the assumption that the companion CPU is running at peak performance and is contributing to the GFLOPS. The FPGA module shows a performance-to-power advantage as it makes good use of the device resources.

Table 6. Matrix Multiplication Performance/Power Ratio

Matrix Multiplier	2 Quad-Core CPUs	CPU + nVidia Tesla S870	CPU + XD2000i w/ 1 EP3S260
Precision	Single	Single	Single
Sustained GFLOPS	140	140	167 (estimated)
Power	~150W	~245W	~110W
GFLOPS/watt	0.9	0.6	1.5

Summary

For many algorithms that contain parallelism or that can be pipelined, the sustained performance of FPGAs can approach peak performance. This is due to the "excess routing bandwidth" that enables a custom datapath, enabling logic to access memory or results from another logic block in one clock. While fixed architectures have a predetermined set of logic blocks for different functions, the FPGA can be configured to have the right ratio of logic functions for a given algorithm for the best use of device resources. The combination of fully utilizing the FPGA resources with near-peak sustained performance results in an excellent performance-to-power ratio.

Great progress is being made in the silicon, arithmetic, and library foundations for FPGAs, leading to strong benchmarking results. FPGA accelerator results will continue to improve over time, and the roadmap is strong for continued performance scaling with future product generations without hitting datacenter power and cooling barriers.

Further Information

1. *The RapidMind Multi-Core Development Platform: Financial Analysis*, Matthew Monteyne, March 2008: www.rapidmind.net/pdfs/WP_Finance.pdf
2. *Designing and Using FPGAs for Double-Precision Floating-Point Math*: www.altera.com/literature/wp/wp-01028.pdf
3. *Floating Point Compiler Increasing Performance With Fewer Resources*: www.altera.com/literature/wp/wp-01050-Floating-Point-Compiler-Increasing-Performance-With-Fewer-Resources.pdf
4. *An evaluation of blade server power efficiency for the HP ProLiant BL260c G5, Dell PowerEdge M600, and IBM BladeCenter HS21 using the SPECjbb2005 Benchmark*, Hewlett Packard, March 2008: ftp://ftp.compaq.com/pub/products/servers/benchmarks/HP_ProLiant_BL260_SPECjbb2005_032808a.pdf
5. *FPGA Acceleration of European Options Pricing*, Nathan Woods, XtremeData Inc., 2008: www.xtremedatainc.com/index.php?option=com_docman&task=doc_download&gid=31&Itemid=129
6. *Accelerating HPC using GPUs*, Glenn Lupton and Don Thulin, Hewlett Packard, May 2008: www.hp.com/techservers/hpccn/hpccollaboration/ADCatalyst/downloads/accelerating_HPC_Using_GPU's.pdf

Acknowledgements

- Martin Langhammer, Principle Scientist, Altera Corporation
- Mike Strickland, Sr. Manger, Strategic and Technical Marketing, Applications Business Group, Altera Corporation

101 Innovation Drive
San Jose, CA 95134
www.altera.com