White Paper

Designing and Using FPGAs for Double-Precision Floating-Point Math

Floating-point arithmetic is used extensively in many applications across multiple market segments. These applications often require a large number of calculations and are prevalent in financial analytics, bioinformatics, molecular dynamics, radar, and seismic imaging, to name a few. Apart from integer and single-precision 32-bit floating-point math, many applications demand higher precision, forcing the use of double-precision 64-bit operations. This paper demonstrates the double-precision floating-point performance of Altera FPGAs using two different approaches. First, a theoretical "paper and pencil" calculation is used to demonstrate peak performance. This type of calculation may be useful for raw comparison between devices, but is somewhat unrealistic as it assumes data is always available to feed the device, and does not take into account memory interfaces and latencies, place and route constraints, and other aspects of an actual FPGA design. Thus, secondly, the real results of a double-precision matrix multiply core that can easily be extended to a full DGEMM benchmark are demonstrated and the real-world constraints and challenges of achieving such results are discussed in detail.

Introduction

An increasing number of applications in many fields, from financial analytics to military radar to various imaging applications, are relying on computations with floating-point (FP) numbers. These applications implement various basic functions and methods such as FFTs, FIR filters, SAR, matrix math, and Monte Carlo. Many of these implementations use single-precision FP, where FPGAs can offer up to ten times the sustained performance compared to CPUs. Recently, there has been increasing interest in double-precision performance to see how well FPGAs can compete with CPUs, especially for designs which have power and cooling constraints.

Dave Strenski's excellent article in HPC Wire, "FPGA Floating-Point Performance—a paper and pencil evaluation," [1] discusses how to estimate the double-precision (64-bit) peak FP performance of an FPGA. This paper evaluates Strenski's method, and, more importantly, expands on it with considerations for estimating the sustained FP performance in an FPGA. These considerations are validated using a matrix multiplication design running in an Altera® Stratix® II EP2S180 FPGA.

In [1], the author references the double-precision general matrix multiply (DGEMM) routine. DGEMM is a common building block for many algorithms and is the most important component of the scientific LINPACK benchmark commonly used on CPUs. The Basic Linear Algebra Subprograms (BLAS) include DGEMM in the Level 3 group. The DGEMM routine calculates the new value of matrix C based on the product of matrix A and matrix B and the previous value of matrix C (α, β are scalar coefficients): $C \leftarrow \alpha AB + \beta C$.

For this analysis, $\alpha = \beta = 1$ is used, though any scalar value can be used as it can be applied during the data transfer in and out. As can be seen, this operation results in a 1:1 ratio of adders and multipliers. This analysis also takes into account the logic required for a microprocessor interface protocol core and adds the following considerations:

- Memory interface module for low latency access to local data
- Data paths from memory interface to FPGA memory
- Data path from FPGA memory to FP cores
- Decrease to FP core F_{MAX} when the FPGA is full
- Unusable FPGA logic due to routing challenges of a full FPGA

The FPGA benchmark focuses on the performance from a real Stratix II EP2S180 implementation of the AB matrix multiplication with data from a locally attached SRAM. The effort to extend this core to include the accumulator to add the old value of C is a relatively minor effort.

WP-01028-1.1

Peak Performance Calculation

The scenario for peak performance uses the same approach used in [1]. Table 1 shows the resources that are available on the EP2S180 FPGA:

Table 1. EP2S180 Resources

Device	ALMs	Equivalent LEs	M512 RAM	M4K RAM	M-RAM	Total Memory Bits	18-bit x 18-bit Multipliers	PLLs
EP2S180	71,760	179,400	930	768	9	9,383,040	384	12

Each ALM contains two 6-input functions (adaptive look-up tables (ALUTs)), with four inputs and two select signals to allow for more efficient use of the logic. Thus, the EP2S180 has 71,760 * 2 = 143,520 ALUTs. In addition, the device has 384 18-bit x 18-bit hardware multiplier/accumulators. The FP adder function uses ALUTs while the FP multiplier uses dedicated multipliers as well as ALUTs from the FPGA fabric. By reserving 22,000 ALUTs for a processor-interface protocol core, 121,520 ALUTs remain for function units.

Table 2 summarizes the resource utilization for the FP cores that are planned for release in 2007.

Table 2. FP Core Resource Utilization

	Synthesis	Multipliers	ALUTs	Latency	Performance
Multiplier	Logic + Multiplier	9	900	13	320 MHz
Adder	Logic	0	1721	17	300 MHz

Using the 1:1 ratio of adders to multipliers, the following 84 double-precision FP cores fit in the device: 42 adder functions using 1721 * 42 = 72,282 ALUTs, and 42 multiplier functions using 42 * 9 = 378 multipliers and 42 * 900 = 37,800 ALUTs. The total resource usage is 110,082 ALUTs out of the available 121,520, with 378 multipliers out of the available 384. Using the slower clock frequency of the two cores yields 300 MHz * 84 FP operations = 25.2 peak giga floating-point operations per second (GFLOPS).

Stratix III Peak Performance Calculation

The peak FP analysis is repeated for the Stratix III EP3SE260 FPGA. Table 3 shows the resources for the EP3SE260, while Table 4 shows the estimated FP core performance in the Stratix III device.

Table 3. EP3SE260 Resources

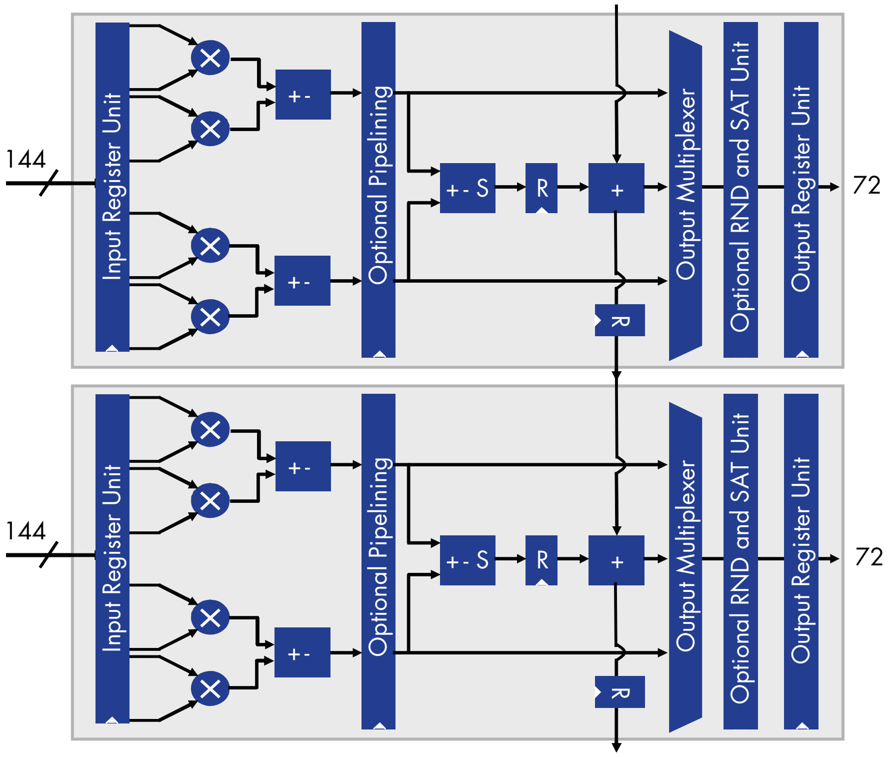
Device	ALMs	Equivalent LEs	Regs	M9K Blocks	M144K Blocks	Embedded Memory (bits)	MLAB (bits)	Max 18-bit x 18-bit Multipliers
EP3SE260	102K	254K	204K	864	40	14.7M	3.3M	768

Table 4. Stratix III FP Core Performance

	Synthesis	Multipliers	ALUTs	Latency	Performance
Multiplier	Logic + Multiplier	9	450	9	380 MHz
Adder	Logic	0	1239	17	330 MHz

When compared to the Stratix II cores, it is apparent that the Stratix III multiplier core is much faster and requires less logic. This is due to innovations in the Stratix III digital signal processing (DSP) block design (shown in Figure 1), including shared inputs to the DSP block. As most functions do not require a 1:1 ratio of routing to logic, sharing inputs allows doubling the density of routing into the block. This increased efficiency means more such blocks can be included on the device and more functionality can be added to the DSP block. Additionally, almost all of the elements of a 54 * 54 multiplication are in the DSP blocks, so only a single external adder is needed.

Figure 1. Stratix III DSP Block



As with the Stratix II FPGA, by using the 1:1 ratio of adds and multiplies and the same resource utilization calculations in a Stratix III FPGA can fit 76 double-precision FP adder functions and 78 double-precision FP multiplier functions to get a very impressive 154 * 330 = 50.16 GFLOPS.

DGEMM Benchmark Code

While peak performance numbers look great on data sheets, most designers also want to know what the sustained performance is with a familiar benchmark. DGEMM is a matrix-matrix multiplication added to an existing value. The product *AB* (matrix *A* multiplied by matrix *B*) is given by $(AB)_{ij} = \sum_{r=1}^{n} a_{ir} b_{rj} = a_{i1}b_{1j} + a_{i2}b_{2j} + \ldots + a_{in}b_{nj}$ for each pair *i* and *j* with $1 \le i \le m$ and $1 \le j \le p$. The DGEMM routine includes adding this product to its previous value.

The C program to implement DGEMM is deceptively simple and includes three nested FOR loops:

```
matrix_multiply()
    {
      for (i = 0 ; i < ROWS ; i++) {
        for (j = 0 ; j < COLUMNS ; j++) {
          for (k = 0 ; k < COLUMNS ; k++) {
            matrix_C[i][j] += matrix_A[i][k] * matrix_B[k][j] ;
          }
        }
      }
    }
```

The benchmark can be accelerated using the parallelism of up to 42 adder and 42 multiplier functions that fit into the EP2S180. The EP2S180 also has more than 42 dual-ported memory blocks, which can be used to stage data from and to local SRAM memory.

Implementation Challenges and Considerations

A single FP core in an FPGA can always achieve the data sheet performance. Of course, this is not representative of any real-world application, where many such cores are used in parallel. Routing multiple 64-bit data paths while completely filling an FPGA with double-precision FP cores is a challenge, usually resulting in unused logic in the device and a decrease in clock speed for some of the FP functions. Since matrix operations require all matrix element calculations to complete at the same time, these parallelized or "vector" operations will occur at the slowest clock speed of all the FP functions in the FPGA. A basic rule of thumb is that the clock speed will degrade by one-third and that 15 percent of the logic in the device will not be used.

Thus, to the peak performance calculation, the following considerations are added to obtain a "constrained performance" prediction:

- Estimated 15 percent logic unusable (due to data path routing, routing constraints, etc.)
- Estimated 33 percent decrease in FP function clock speed
- Extra 24,000 ALUTs for local SRAM memory controller and processor interface

With those assumptions, only 39 adders and 39 multipliers will fit in an EP2S180 and the clock speed will drop from 300 MHz to 200 MHz, yielding a predicted sustained performance of 15.7 GFLOPS (compared to the peak performance of 25.2 GFLOPS).

In the benchmark conducted, the values for the *A* and *B* matrices are transferred from the microprocessor memory to SRAM, which is locally attached to the EP2S180 FPGA. This benchmark is focused on the capabilities of the EP2S180 FPGA with local external SRAM, so the latency involved with the DMA transfer of the *A* and *B* matrix values from microprocessor to local FPGA SRAM and back will not be included in the benchmark time.

Another challenge in using all the double-precision FP cores that fit in the FPGA is feeding them all with data on every clock cycle. When dealing with double-precision 64-bit data, and parallelizing many FP arithmetic cores, wide internal memory interfaces are needed. The flexible use of the various memory structures within the Stratix II and III devices allows for optimization of this on-chip Level 1 cache.

Benchmark Results

A double-precision FP matrix multiplication core designed at Altera is used with an application program interface (API)/library call for higher level tools such as Impulse C. In this design, 40 double-precision multiplier cores are used as well as 45 double-precision adder cores for an adder tree and blocking implementation. (Blocking is a common technique that splits a matrix into smaller matrices or blocks to allow optimal usage of available cache). The memory approach puts sub-blocks of *A* into MRAM (4K * 144 bits) memory and sub-blocks of *B* into M4K (128 * 36 bits) memory. The resulting hardware runs at 200 MHz, with enough logic remaining to allow for the CPU interface and memory cores. Table 5 shows a few processing matrix sizes (actual matrix sizes are much larger) in order to evaluate the time added to read and write the date from an SRAM interface of 4 bytes per clock cycle (800 Mbps bandwidth).

Table 5. Processing Matrix Sizes (Blocks)

A	B	C
128 x 160	160 x 64	128 x 64
170 x 120	120 x 85	170 x 85
100 x 160	160 x 10	100 x 10

The average sustained throughput was 88 percent. The 40 multiply and 40 adder tree cores generate a result every clock cycle, while the five additional adder cores used for blocking implementation together average about one value per clock cycle. The GFLOPS calculation then is 200 MHz * 81 operators * 88 percent duty cycle = 14.25 GFLOPS. This is lower than the constrained performance prediction of 15.7 GFLOPS, due mainly to the time needed to read and write values to the external SRAM. With multiple SRAM banks providing higher memory bandwidth, the GFLOPS would be closer to the 15.7 GFLOPS number.

An encrypted version of this core is available to those who may want to duplicate these results. One observation of this benchmark is that results will differ slightly between an algorithmic approach of sequential accumulation per value of *C*, as opposed to using an adder tree. The choice of adder tree allows for greater flexibility in matrix sizes, while the sequential accumulation approach works well with larger matrices. Another difference between the two approaches relates to accuracy of the result, but the topic of bit accuracy versus the Pentium at the algorithmic level is out of the scope of this treatise and is worthy of a separate paper.

The matrix multiply core can easily be extended to accumulate against a pre-existing matrix *C*, and to include the two scalar multiplications (for α and β) to make it a full DGEMM implementation. Although this extension will not consume many FPGA resources, more SRAM bandwidth is required. The resulting GFLOPS for this extension should be close to the ones presented for the matrix multiply.

GFLOPS/Watt

As FPGAs can be completely dedicated to a particular function, they are more energy efficient than general-purpose CPUs which must be able to handle a variety of functions, and which support a large memory subsystem with different layers of cache.

Recent research from The Green Grid consortium indicates that a typical 2U server with dual processors consumes 450 W of power. In addition, Gartner analysts say traditional datacenters typically waste more than 60 percent of the energy they use to cool equipment, so it will take more energy to cool the datacenter. Gartner has stated that 50 percent of datacenters worldwide will max out their available electrical power by the end of 2008. In some urban datacenters, there is no room left for air-conditioning on the office building rooftops, so power consumption is a serious concern.

An energy and cooling capacity-constrained datacenter with scientific workloads similar to DGEMM should consider looking at FPGA technology for application acceleration. The expected 15 GFLOPS performance of the Stratix EP2S180 FPGA running at 30 W is close to the sustained performance of a 60-W 3-GHz Intel Woodcrest CPU, thus the FPGA has a very high performance per Watt ratio.

Conclusion

As FPGAs were originally designed as reconfigurable logic devices for rather simple combinatorial and sequential operations, they were not well suited for FP arithmetic operations and even less suited for double-precision FP operations. However, over the last few years FPGAs have matured into high-end compute engines that can rival any traditional CPU or digital signal processor in raw GFLOPS. FPGAs are register-rich devices that allow for massive pipelining and stringing of operators together to produce results on every clock cycle. Designed for optimal double-precision FP operations, the latest Stratix III FPGAs can implement many computational elements in parallel for a given function and thereby deliver impressive performance.

The matrix multiplication design described in this paper takes advantage of these inherent FPGA attributes. While the peak FP performance calculations of [1] are a good beginning, the constrained performance analysis of an FPGA provides the veracity of implementing such a design in a real-world scenario. In the real-world, data are read and written to and from external memory, and many 64-bit wide data paths must be routed within the FPGA. In addition, achieving optimal performance with FP operations can be challenging, as it is critical to understand all the design constraints that will eventually degrade the data sheet performance numbers. The effect of these constraints on design and performance of a common benchmark is demonstrated using the real design of a double-precision matrix multiplier targeted to and run in an Altera Stratix II FPGA.

Further Information

1. Dave Strenski, “FPGA Floating-Point Performance—a paper and pencil evaluation,” HPC Wire, January 12, 2007:
   www.hpcwire.com/hpc/1195762.html

2. “The Green Grid Opportunity, Decreasing Datacenter and Other IT Energy Usage Patterns,” The Green Grid Consortium, February 16, 2007:
   www.thegreengrid.org/gg_content/Green_Grid_Position_WP.pdf

3. “Gartner Says 50 Percent of Data Centers Will Have Insufficient Power and Cooling Capacity by 2008,” Gartner Inc. Press Release, November 29, 2006:
   www.gartner.com/it/page.jsp?id=499090

Acknowledgements

- Danny Kreindler, Sr. Strategic Marketing Manager, Applications Business Group, Altera Corporation
- Mike Strickland, Director, Strategic and Technical Marketing, Applications Business Group, Altera Corporation