WHITE PAPER

FPGA

Understanding Peak Floating-Point Performance Claims

Learn how to calculate and compare the peak floating-point capabilities of digital signal processors (DSPs), graphics processing units (GPUs), and FPGAs.

Authors

Michael Parker
Principal DSP Planning Manager
Intel Programmable Solutions Group

Table of Contents

Introduction

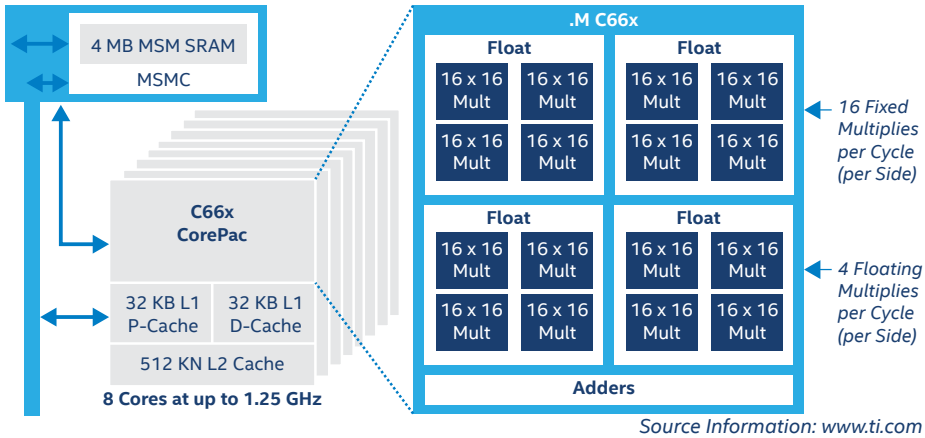
DSPs, GPUs, and FPGAs serve as accelerators for the CPU, providing both performance and power efficiency benefits. Given the variety of computing architectures available, designers need a uniform method to compare performance and power efficiency. The accepted method is to measure floating-point operations per second (FLOPS), where a FLOP is defined as either an addition or multiplication of single (32 bit) or double (64 bit) precision numbers in conformance with the IEEE 754 standard. All higher order functions, such as divide, square root, and trigonometric operators, can be constructed using adders and multipliers. As these operators, as well as other common functions such as fast Fourier transforms (FFTs) and matrix operators, require both adders and multipliers, there is commonly a 1:1 ratio of adders and multipliers in all these architectures.

DSP, GPU, and FPGA performance comparison

We compare the performance of the DSP, GPU, and FPGA architectures based on their peak FLOPS rating. The peak FLOPS rating is determined by multiplying the sum of the adders and multipliers by the maximum operation frequency. This represents the theoretical limit for computations, which can never be achieved in practice. It is generally not possible to implement useful algorithms that can keep all the computational units occupied all the time. It does however provide a useful comparison metric.

DSP peak GFLOPS

An example is provided for Texas Instruments' TMS320C667x DSP. This DSP contains eight DSP cores, with each core containing two processing subsystems.



Source Information: www.ti.com

Figure 1. TMS320C667x DSP Architecture

Each subsystem contains four single-precision floating-point adders and four single-precision floating-point multipliers. This is a total of 64 adders and 64 multipliers. The fastest version available runs at 1.25 GHz, providing a peak of 160 GigaFLOPS (GFLOPS).

GPU peak GFLOPS

One the most powerful GPUs is the NVIDIA Tesla K20. This GPU is based upon CUDA cores, each with a single floating-point multiple-adder unit, which can execute one per clock cycle in single-precision floating-point configuration. There are 192 CUDA cores in each Streaming Multiprocessor (SMX) processing engine. The K20 actually contains 15 SMX engines, although only 13 are available (for example, due to process yield issues). This gives a total of 2,496 available CUDA cores, with 2 FLOPS per clock cycle, running at a maximum of 706 MHz. This provides a peak single-precision floating-point performance of 3,520 GFLOPS.

FPGA peak GFLOPS

Intel® offers hardened floating-point engines in their FPGAs. A single-precision floating-point multiplier and adder have been incorporated into the hard DSP blocks embedded throughout the programmable logic structures. A medium-sized, midrange Intel Arria® 10 FPGA is the 10AX066. This device has 1,678 DSP blocks, each of which can perform 2 FLOPS per clock cycle, resulting in 3,376 FLOPS each clock cycle. At a rated speed of 450 MHz (for floating point; the fixed-point modes are higher), this provides for 1,520 GFLOPS. Computed in a similar fashion, Intel states that up to 10,000 GFLOPS, or 10 TeraFLOPS, of single-precision performance are available in the high-end Intel® Stratix® 10 FPGAs, achieved with a combination of both clock rate increases and larger devices with much more DSP computing resources.

Floating-point in FPGAs using programmable logic

Floating point has always been available in FPGAs using the programmable logic of the FPGA. Furthermore, with programmable logic based floating point, an arbitrary precision level can be implemented, and is not restricted to industry-standard single and double precision. Intel offers seven different levels of floating-point precision. However, determining the peak floating-point performance of a given FPGA using programmable logic implementation is not at all straightforward. Therefore, the peak floating-point rating of Intel FPGAs is based solely on the capability of the hardened floating-point engines, and assumes that the programmable logic is not used for floating point, but rather for the other parts of a design, such as the data control and scheduling circuits, I/O interfaces, internal and external memory interfaces, and other needed functionality.

Challenges to determine floating-point performance

There are several factors that make the calculation of floating-point performance using programmable logic very difficult. The amount of logic to build one single-precision floating-point multiplier and adder can be determined by

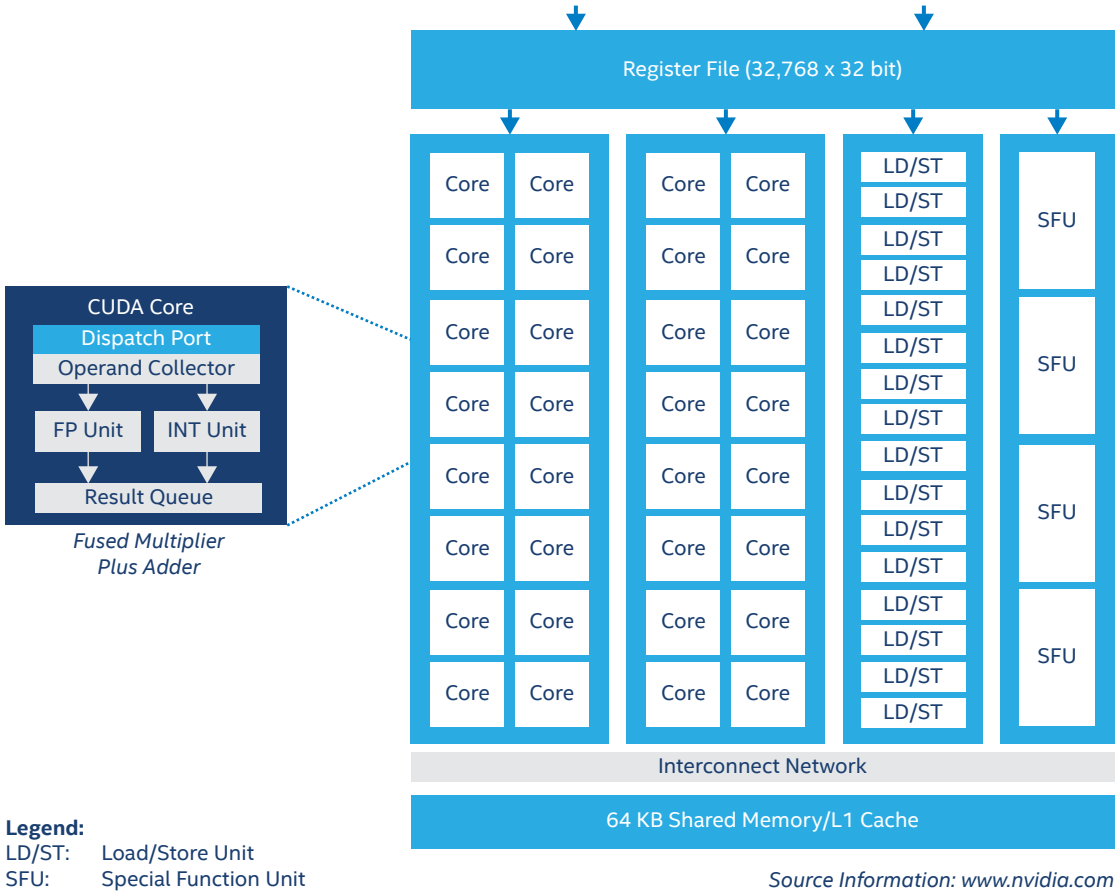


Figure 2. GP-GPU Architecture

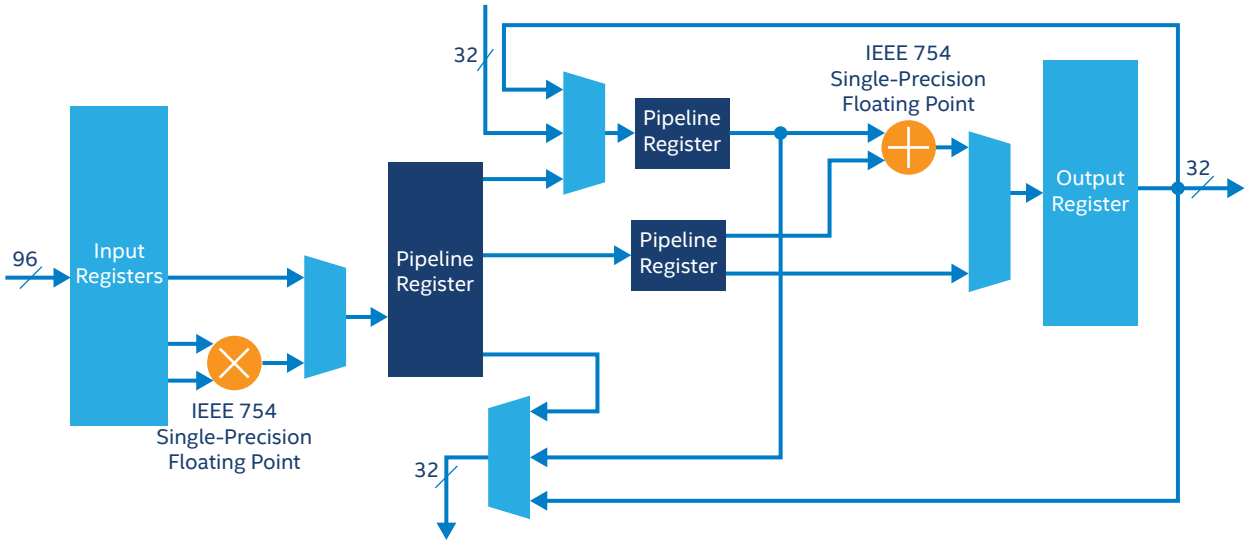


Figure 3. Floating-Point DSP Block Architecture in FPGAs

consulting the FPGA vendor's floating-point intellectual property (IP) user guide. However, one vital piece of information is not reported in the user guide, and that is the routing resources required. To implement floating point, large barrel shifters, which happen to consume tremendous amounts of the programmable routing (interconnect between the programmable logic elements), are required. All FPGAs have a given amount of interconnect to support the logic, which is based on what a typical fixed-point FPGA design will use. Unfortunately, floating point does require a much higher degree of this interconnect than most fixed-point designs. When a single instance of a floating-point function is created, it can draw upon routing resources in the general region of the logic elements used. However, when large numbers of floating-point operators are packed together, the result is routing congestion. This causes a large reduction in achievable design clock rates as well as logic usage which is much higher than a comparable fixed-point FPGA design. Intel has a proprietary synthesis technique known as "fused datapath" which does mitigate this to some extent, and allows very large floating-point designs to be implemented in the logic fabric, and leveraging fixed-point 27 x 27 multipliers for single precision, or 54 x 54 for double precision.

In addition, the FPGA logic cannot be fully used. As the design takes up a large percentage of the available logic resources, the clock rate or f_{MAX} at which timing closure can be achieved is reduced, and eventually timing closure cannot be achieved at all. Typically, 70 - 90% of the logic can actually be used, and with dense floating-point designs, it tends to be at the lower end of this range.

Benchmarking floating-point designs

Due to these reasons, it is nearly impossible to calculate the floating-point capacity of an FPGA when implemented in programmable logic. Instead, the best method is to build benchmark floating-point designs, which include the timing closure process. Alternatively, the FPGA vendor can supply such designs, which would greatly aid in estimating what is possible in a given FPGA.

Intel provides benchmark designs on 28 nm FPGAs, which cover basic as well as complex floating-point designs. The published results show that with 28 nm FPGAs, several hundred GFLOPS can be achieved for simpler algorithms such as FFTs, and just over 100 GFLOPS for complex algorithms such as QR and Cholesky decomposition.

- For the benchmark results, refer to the Radar Processing: FPGAs or GPUs? (PDF) white paper.

In addition, Berkeley Design Technology, Inc. (BDTI), a third-party technology analysis firm, conducted an independent analysis on complex, high-performance floating-point DSP designs on Intel's 28 nm FPGAs.

- For information from this independent analysis, refer to the An Independent Analysis of Floating-point DSP Design Flow and Performance on Intel 28-nm FPGAs (PDF) white paper by BDTI.

Many other floating-point benchmark designs have also been implemented using OpenCL™ [§] on Intel's 28 nm Stratix V FPGAs, and are available upon request. These designs are in the process of migrating to the Arria 10 FPGAs, which will provide dramatic performance improvements due to the hardened floating-point DSP block architecture.

How not to calculate FPGA GFLOPS

To reiterate, floating point has always been possible in FPGAs, implemented largely in programmable logic and using fixed-point multiplier resources. But the degree of ambiguity in actual performance and throughput can allow for some very aggressive marketing claims. We present a real-life example of how such an artificial benchmark could be constructed using Xilinx's 28 nm and 20 nm FPGAs as the case study.

Refer to Table 1 for resource information by Xilinx to construct a single floating-point function and determine the performance for the Kintex-7 FPGA, built on the 28 nm process.

[§] OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission by Khronos

Characterization of Single-Precision Format on Kintex-7 FPGAs

Operation	Resources						Maximum Frequency (MHz)
	Embedded		FPGA Logic			18K Block RAMs	Kintex-7
	Type	Number	LUT-FF Pairs	LUTs	FFs		-1 Speed Grade
Add/Subtract	DSP48E1 (speed optimized, full usage)	2	354	238	342	0	463
	Logic (speed optimized, no usage)		578	379	602	0	550
	Logic (low latency)		656	500	668	0	447
Fused Multiply-Add	DSP48E1 (full usage)	4	1,283	802	1,233	0	488
	DSP48E1 (medium usage)	2	1,246	766	1,238	0	438
Accumulator	DSP48E1 (full usage)	5	1,537	894	1,551	0	410
	DSP48E1 (medium usage)	2	1,566	915	1,588	0	412
	Logic		1,662	1,097	1,552	0	375
Fixed to float	Int32 input		213	164	229	0	559
Float to fixed	Int32 result		242	175	237	0	557
Float to float	Single to double		70	21	70	0	625
Compare	Programmable		51	48	12	0	624
Divide	RATE = 1		1,252	777	1,656	0	425
	RATE = 26		256	209	209	0	406

Source Information: www.xilinx.com

Table 1. Xilinx Floating-Point Functions Resources and f_{MAX}

Note: The Virtex-7 device shares the same core architecture as the Kintex-7 device, so the numbers in Table 1 can be used as reference for our example that follows.

The largest logic density 28 nm device offered by Xilinx is the Virtex-7 XC7V2000T device, with 1,954.56K logic cells (LCs) and 2,160 DSP slices (with 18 x 25 fixed-point multipliers).

If the fused multiply-add function was chosen for benchmarking, only 1,440 units could be built due to the scarcity of DSP slices. Also, the rated speed is 438 MHz for a single instantiation in the fastest speed device, making this a poor choice for a peak GFLOPS claim, although the multiplier-adder function is the standard architecture used in floating-point architectures.

To maximize the floating-point rating, the best choice is to use the add/subtract function. The best strategy is to build as many adders as possible until the DSP48E slices are exhausted, and build the remaining adders with pure logic.

The DSP resourced adders require 2 DSP slices and 354 LUT-FF pairs (or LCs) and a single instantiation can operate at 463 MHz. The logic-based adder uses 578 LCs, and a single instantiation can operate at 550 MHz.

We assume 100% logic and 100% DSP slices are used (this requires enough routing to be available to utilize all of the logic).

Using this method, the Xilinx could claim 1,996 GFLOPS of single-precision floating-point performance.

Similarly, at 20 nm, the largest logic density device offered by Xilinx is the UltraScale XCVU440 device. It features 4,407.48K LCs and 2,880 DSP slices (each of which supports 18 x 27 fixed-point multipliers). Table 1 can also be used for 20 nm Ultrascale devices, as the device core architectures are very similar. The 20 nm process will also offer an increase in maximum frequency, assumed at 12%.

Number of Adders	Adder Type and f_{MAX}	f_{MAX}	GFLOPS	LCs per Adder	LCs Required	DSP Slices per Adder	DSP Slices Required
1,080	DSP based	463 MHz	500.04	354	382.3K LE	2	2,160
2,720	Logic based	550 MHz	1,496	578	1,572.2K LE	0	0
Total							
3,800			1,996		1,954.5K LE		2,160

Table 2. Calculated Peak GFLOPS on 28 nm Xilinx FPGA

Number of Adders	Adder Type and f_{MAX}	f_{MAX}	GFLOPS	LCs per Adder	LCs Required	DSP Slices per Adder	DSP Slices Required
1,440	DSP based	518.56 MHz	746.7	354	509.8K LEs	2	2,880
6,743	Logic based	616 MHz	4,157.7	578	3,897.4K LEs	0	0
Total							
8,183			4,904		4,407.2K LEs		2,880

Table 3. Calculated Peak GFLOPS on 20 nm Xilinx FPGA

Just coincidentally, this number is what Xilinx claims, although the method used to determine the GFLOPS is not provided.

There is a reason for the 1:1 ratio of floating-point multipliers and adders in most computer architectures. Applications using multipliers include matrix multiplication, FFTs, higher order mathematical functions, such as square root, trigonometric, and log, matrix inversion, and all linear algebra intense designs. Any useful floating-point application requires a large number of floating-point multipliers; hence the practical reason why computer architectures offer equal numbers of multipliers and adders. Although high ratings can be claimed by using only floating-point adders (no multipliers), such designs have no application benefits.

As every experienced FPGA designer knows, the design f_{MAX} decreases as the percentage of logic used increases. Generally, logic usage of above 80% results in significant degradation in the clock rate at which timing closure can be achieved. Furthermore, a large amount of logic is required for other functions, which further reduce the available logic for floating-point computations.

The choice of FPGA family is also significant. The family chosen is very high-cost devices, generally used for ASIC prototyping applications, where device cost and power consumption are not important factors. Using the FPGA family optimized for DSP applications would dramatically lower GFLOPS calculations.

While not demonstrated here, early power estimators are available from FPGA vendors. A design using all of the resources and f_{MAX} defined in Table 3 would have power consumption in the hundreds of watts. This particular FPGA is intended for ASIC prototyping applications, where clock rates and toggle rates are low. High clock rates and toggle rates will dramatically increase power consumption beyond the VCC current and power dissipation capabilities of the device.

In short, this is not a recognized method of benchmarking by the industry, and hence, the numbers obtained using this method should not be used to compare with the peak floating-point performance claims from other semiconductor vendors. The peak GFLOPS figure should represent the achievable performance of a given device. Of course, some derating is still needed, as almost no algorithm can keep all the computational units performing useful calculations at 100% duty cycle.

Conclusion

FPGAs with hardened floating-point DSP blocks are now available, and provide single-precision performance from 160 to 1,500 GFLOPS in midrange Arria 10 devices, and up to 10,000 GFLOPS in high-end Stratix 10 devices. These peak GFLOPS metrics are computed based on the same transparent methodology used on CPUs, GPUs, and DSPs.

This methodology provides designers a reliable technique for the baseline comparison of the peak floating-point computing capabilities of devices with very different architectures. The next level of comparison should be based on representative benchmark designs implemented on the platform of interest.

For FPGAs lacking hard floating-point circuits, using the vendor-calculated theoretical GFLOPS numbers is quite unreliable. Any FPGA floating-point claims based on a logic

Maximum DSP Capabilities	Zynq-7000	7 Series	UltraScale
Logic Cells	444K	1,955K	4,400K
Block RAM	3,020 Kb	68 Mb	115 Mb
DSP Slices	2,020	3,600	5,500
Fixed-Point Performance	2,622 GMACS	5,335 GMACS	8,151 GMACS
Floating-Point Performance	778 GFLOPs	2,000 GFLOPs	4,903 GFLOPs
Transceivers	16	96	104
Transceiver Performance	10.3123 Gbps	12.5 Gbps 13.1 Gbps 28.05 Gbps	16.3 Gbps 32.75 Gbps

Source Information: www.xilinx.com

Please note the assumptions used in this calculation:

- Only floating-point adders, and no floating-point multipliers were used
- 100% of the FPGA logic were used
- Timing for a complete design was closed at the f_{MAX} of a single floating-point operator
- Zero logic resources were available for memory interfacing, control circuits, etc.
- Devices chosen were not the DSP-rich members of the FPGA family, but rather a device useful for emulating ASIC logic as a prototype platform

Table 4. Xilinx Floating-Point Performance Claims

implementation at over 500 GFLOPS should be viewed with a high level of skepticism. In this case, a representative benchmark design implementation is essential to make a comparative judgment. The FPGA compilation report showing logic, memory, and other resources along with the achieved clock rate, should also be provided. Better yet, any party claiming a specific performance level should be willing to make available the compiled design file, allowing the results to be replicated.

Where to get more information

For more information about Intel and Arria 10 FPGAs, visit **https://www.altera.com/products/fpga/arria-series/arria-10/overview.html**

[1] http://www.altera.com/literature/po/bg-floating-point-fpga.pdf
[2] http://www.altera.com/literature/wp/wp-01197-radar-fpga-or-gpu.pdf
[3] http://www.altera.com/literature/wp/wp-01187-bdti-altera-fp-dsp-design-flow.pdf