WP409 (v1.0) October 31, 2011

High-Level Implementation of Bit- and Cycle-Accurate Floating-Point DSP Algorithms with Xilinx FPGAs

By: Tim Vanevenhoven

Floating-point arithmetic, long the realm of general-purpose CPUs, DSPs, and graphics processing units (GPUs) is seeing growing use in FPGAs. This trend is driven by a host of new applications in medical imaging, wireless, and defense that require large dynamic range as well as by a strong need to simplify the design process.

Xilinx 7 series FPGAs can deliver up to 1.33 teraflops of single-precision floating-point performance on one device, driving the demand for an easy-to-use design flow that delivers hand-crafted results. Xilinx System Generator for DSP™ now meets this demand by supporting the design and implementation of floating-point algorithms from within the MathWorks Simulink modeling environment. System Generator also has the flexibility of optimizing an implementation that is bit- and cycle-accurate to the original model.

The Need for Floating-Point Implementations

Xilinx FPGAs have long been used to implement fixed-point DSP and video algorithms in hardware. The flexibility of programmable logic allows fixed-point arithmetic to use custom bit widths that are not bound to the 8-, 16-, or 32-bit boundaries of a fixed-point processor. Fixed-point bit widths can grow as needed to accommodate applications that require larger dynamic range. However, as the dynamic range needs grow, a fixed-point implementation becomes increasingly expensive.

7 series FPGAs include an optimized DSP block called the DSP48E1 slice that includes a 25x18 multiplier (Figure 1). Five high-speed interconnects connect two DSP48E1 slices into a single DSP48E1 tile that supports a single-precision floating-point multiply with no loss of F_{MAX} performance. The Virtex®-7 family, the largest of the 7 series FPGAs, includes 1800 DSP48E1 tiles (3600 DSP48E1 slices) and can deliver up to 1.33 teraflops of single-precision DSP performance on one device at a fraction of the cost and power of GPUs. This makes Virtex-7 FPGAs an attractive choice for cost- and power-sensitive applications that require floating-point hardware.

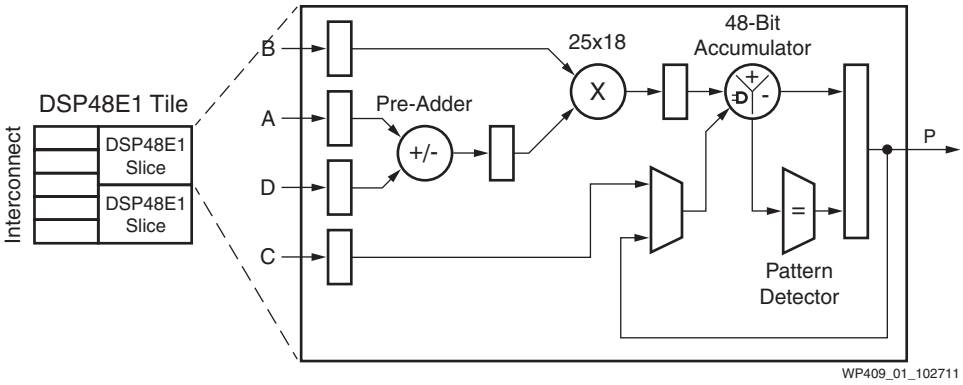

Figure 1: **DSP48E1 Tile and DSP48E1 Slice**

Engineers and scientists developing new algorithms look for efficient ways to implement floating-point algorithms in FPGAs without using hardware description languages like VHDL or Verilog. The design flow for fixed-point implementation is well known, but there is a gap in floating-point design flows that require abstraction along with the capability to fine tune an implementation to meet challenging system requirements. Hardware description languages provide an efficient design flow for fixed-point arithmetic because synthesis tools can directly produce efficient fixed-point hardware for arithmetic operators such as * (multiplication) and + (addition). Floating-point hardware design, however, relies heavily on the use of instantiated IP cores such as the Xilinx floating-point operator (FPO), which makes the use of HDL cumbersome and simulation difficult.

Precision Considerations

While the IEEE 754 standard specifies three basic floating-point formats, this white paper touches on two of them—single and double precision. Single precision uses a total of 32 bits, including one sign bit, eight exponent bits, and 23 fractional bits. Double precision uses a total of 64 bits, including one sign bit, 11 exponent bits, and 52 fractional bits. It is appropriate to only support the IEEE 754 single and double precision bit widths for CPU, DSP, and GPU processors that include dedicated floating-point instruction sets. Conversely, the flexibility of programmable logic allows for hardware savings by customizing the size of the floating-point exponent or mantissa fields to the exact precision requirements for the application. For example, a particular wireless system might only need 28 fractional bits to maintain sufficient fidelity. With System Generator, this can be specified on a per-block basis and quickly simulated in Simulink to observe the effects (Figure 2).

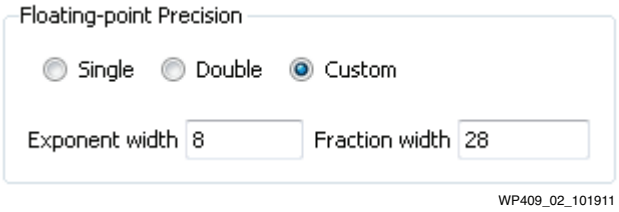

Figure 2: **System Generator Floating-Point Precision Options**

The resulting design provides sufficient numeric accuracy while requiring less logic and power than a competing implementation in which the designer would be forced to use the full 64-bit IEEE 754 double precision.

Automatic Data Type Propagation

System Generator further simplifies the design flow by supporting automatic floating-point data type propagation. With floating-point operations, the result of an operation is stored using the same floating-point precision. There is no concept of bit growth as found in fixed-point operations. The exponent field in the operation manages the need for increased dynamic range. If the user specifies custom floating-point bit widths for the exponent and mantissa, this data type is automatically propagated throughout the design (Figure 3).

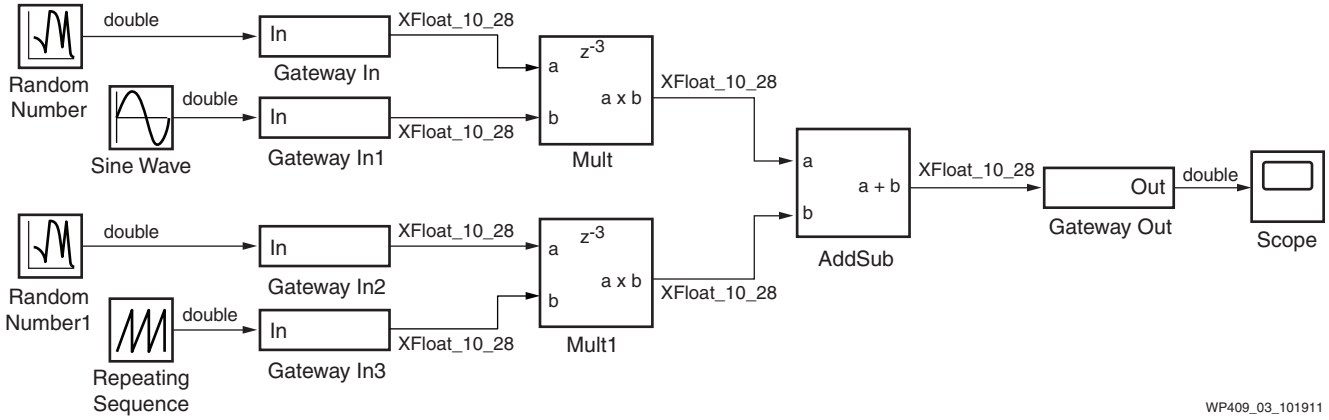

Figure 3: **Floating-Point Data Type Propagation in System Generator**

The System Generator floating-point library contains over 30 blocks. For blocks that support both floating- and fixed-point data, the implementation is determined by the data type of the block's inputs. This makes the design flow simple and easy to understand because there is no need to swap out blocks when switching some or all of the design between a fixed- or floating-point implementation during the design cycle.

Bit- and Cycle-Accurate Verification

One of the challenges with alternative floating-point design flows is verification. Even when only using single- and double-precision data types, the algorithmic simulation and implementation in hardware do not always match. This could be because the algorithmic simulation might be using a higher internal precision or because the normalization logic is not the same as was used in the simulation. With the System Generator design flow, bit accuracy is not a concern. For all designs, whether containing single, double, or custom precision data, the implementation is bit- and cycle-accurate to the original design source. Accordingly, when implementing a design that leverages custom precision operators and datapaths, the implementation is bit- and cycle-accurate to the original simulation model. In addition, the hardware co-simulation capabilities available in System Generator also fully support floating-point implementations for accelerated verification in hardware.

Conclusion

Up until recently, most algorithms implemented in FPGAs were fixed-point. Current trends in system requirements and available FPGAs are causing floating-point implementations to become more common. The IEEE 754 standard specifies data types that are sufficient for fixed-architecture processors but limit the flexibility available in FPGAs. Using only these data types unnecessarily restricts designers from optimizing their floating-point FPGA implementation.

The high-level floating-point design flow available today with System Generator provides users a powerful environment that allows the creation of custom precision datapaths for optimal area and power. Unlike alternative FPGA design flows, the floating-point design flow also generates an implementation that is bit- and cycle-accurate to the original simulation model. To learn more about this design flow, go to www.xilinx.com/system_generator.

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
10/31/11	1.0	Initial Xilinx release.

Notice of Disclaimer