

Extending DSP Design to Heterogeneous Hardware Platforms

You can add an FPGA co-processor to a DSP system in an automated flow.

by Tom Hill
System Generator Product Manager
Xilinx, Inc.
tom.hill@xilinx.com

Standards in the video, imaging, and telecommunications markets have driven the use of heterogeneous, reconfigurable DSP hardware platforms. In the context of this article, these platforms include both DSP processors and FPGAs. They provide an off-the-shelf hardware solution that addresses the most important design challenges for video, imaging, and telecommunications, yet are still sufficiently customizable to allow for product differentiation.

In 2005, market research firm Forward Concepts published a survey (Figure 1) that concluded that the main selection criteria for both processors and FPGAs is not the devices themselves but the tools for developing them. This same concept should hold true for platforms that include an FPGA and DSP processor.

Traditional DSP developers often select DSP processors over FPGAs because the design flow is known and the benefits of a heterogeneous system are difficult to evaluate. Reconfigurable hardware platforms limit degrees of freedom in the hardware, and in doing so allow for greater automation in the design flow. Through this automation, complexity can be eliminated, bringing the benefits of hardware solutions to an extended segment of the DSP design community.

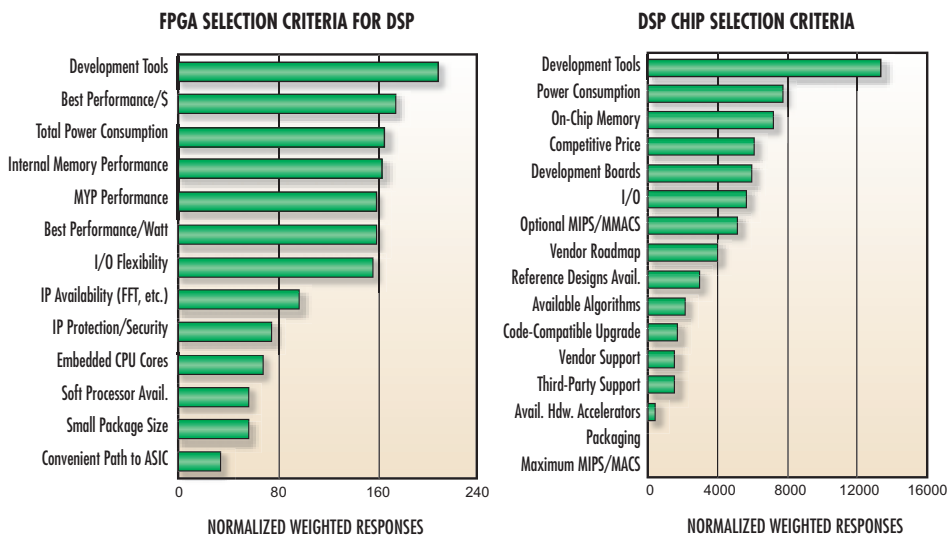


Figure 1 – 2005 Forward Concepts market survey

Benefits of DSP Hardware Platforms

FPGAs and DSP processors have fundamentally different architectures. An algorithm that is well suited for implementation on one device may be very inefficient on the other. A hardware system based solely on DSP processors may require more area, cost, or power if the target application requires a large amount of parallel processing or a maximized multi-channel throughput. An FPGA co-processor can provide as many as 550 parallel multiply and accumulate operations on a single device, delivering the same performance with fewer devices and lower power for many applications (Figure 2).

Although FPGAs excel at processing large amounts of data in parallel, they are not as optimized as processors for tasks such as periodic coefficient updates, decision-making control tasks, or high-speed serial mathematical operations. It is the combination of both the FPGA and DSP processor that delivers winning solutions for a wide range of applications.

For example, a heterogeneous, reconfigurable DSP platform can be ideal for smart cameras that employ pattern recognition. The parallel processing capacity of the FPGA is well suited to image segmentation and feature extraction,

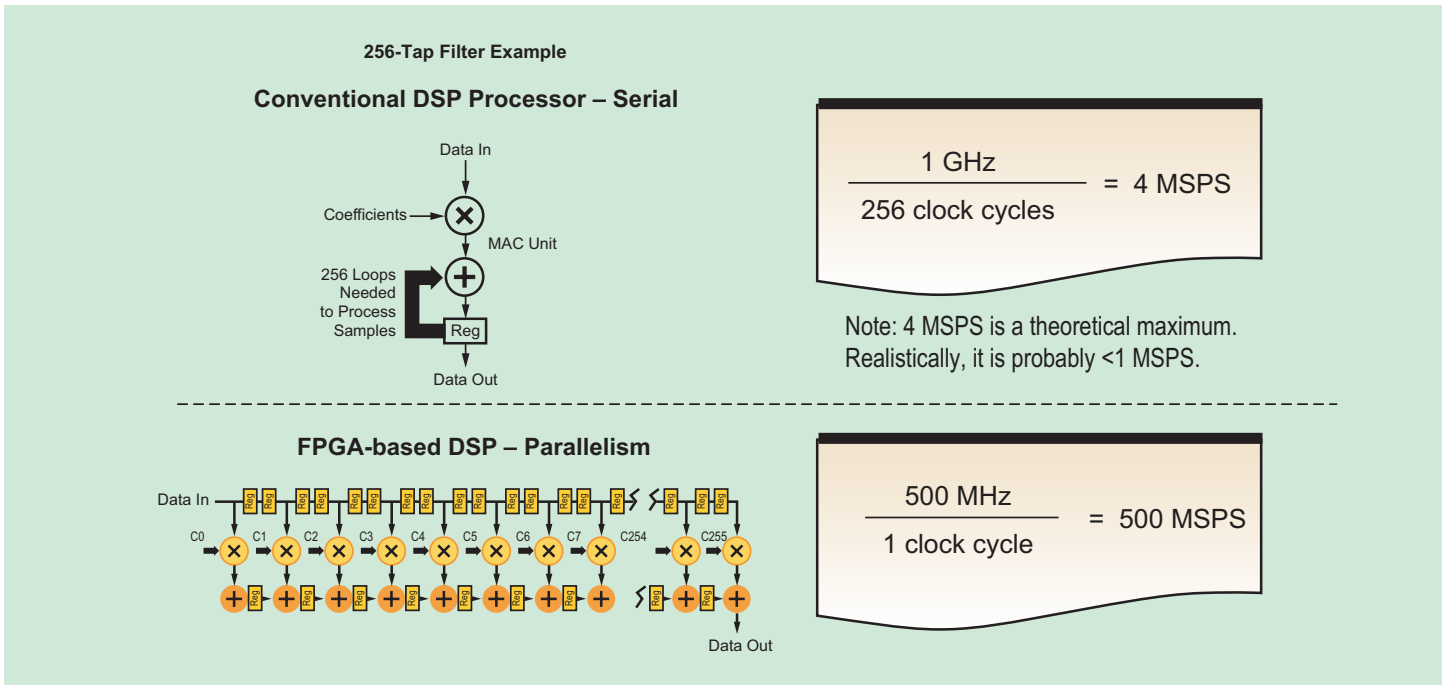


Figure 2 – FPGA-based DSP – parallelism

while a video and imaging DSP processor is better suited to math-intensive tasks such as statistical pattern classification. A heterogeneous system allows better exploitation of pipelining and parallel processing, which are essential to achieve high frame rates and low latency.

Benefits of a Heterogeneous Platform-Based Design Flow

Heterogeneous platform-based design flows extend those design automation concepts adopted by the individual processor and FPGA design flows to the entire platform. The basic tenets of platform-based design are to abstract away the “middleware” of hardware- and software-based systems. This allows DSP designers with little or no FPGA design experience to evaluate and exploit the performance, cost, and power benefits of an FPGA coprocessor.

A platform-based design flow should automatically generate memory maps; header and driver files for the software interface; and hardware interface and interrupt logic. Refinement of the overall system should have limited consequences to the individual hardware and software components (Figure 3).

Through this automation, a single developer no longer needs to master the broad

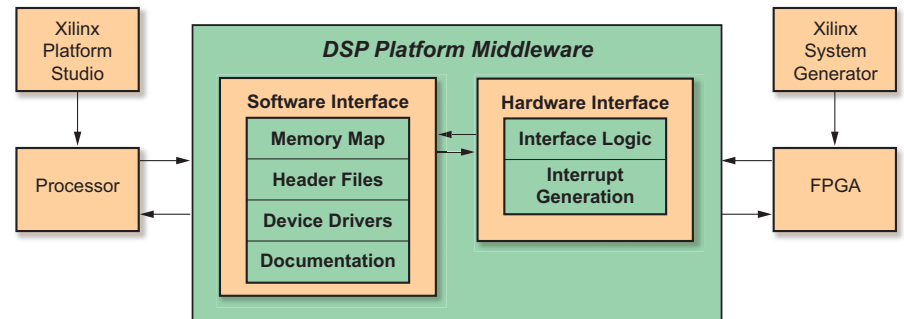


Figure 3 – Hardware/software interface generation

range of technologies required to design FPGA hardware, DSP processor application code, and interface logic and software.

Designing an FPGA Coprocessor

There are many ways to implement a signal processing algorithm in any given technology. The algorithmic approach is often influenced by the target hardware. When the target is a heterogeneous DSP hardware platform, the selection of an implementation becomes a two-step process. You must first select the most appropriate hardware device and then

determine which implementation method makes sense for that device.

On a reconfigurable DSP hardware platform, the processor will be the master and control the FPGA. The FPGA, in turn, will be used as either a coprocessor (where data is sourced to and synched from the DSP processor), or as a pre- or post-processor (where the data is sourced from a high-speed interface). The optimal usage of the FPGA is driven from the system data rates, format, and operating parameters.

Tools such as Code Composer Studio for Texas Instruments DSPs include code

profilers that identify the software “hot spots” that can be offloaded to the FPGA. It is not uncommon for 20% of the application code to consume 80% of the available processor MIPS.

An interface will be required to connect the FPGA to a separate DSP processor on the hardware platform. Reconfigurable DSP platforms will typically support more general-purpose interfaces such as the Texas Instruments 16-/32-/64-bit Tic6x DSP extended memory interface (EMIF) (suitable for system control and coprocessing tasks) and high-speed serial interfaces such as SRIO or video interfaces (for pre- and post-processing operations).

As FPGA coprocessors are added to the system, the software implementation will change from an algorithmic description to

and accelerate your Simulink simulations.

System Generator now supports platform-based design through automatic generation of the infrastructure between an FPGA coprocessor and a Texas Instruments DSP processor. This support is platform-specific and initially offered for the Xilinx Video Co-Processing Kit. Future releases of System Generator will include support for additional platforms.

With this new automation, System Generator provides an abstraction layer between the hardware and software through special blocks called “shared memories.” To the hardware developer, this shared memory behaves the same as a single port of FIFO, RAM, or register (Figure 4).

Passing data to and from the FPGA is accomplished through a simple function call

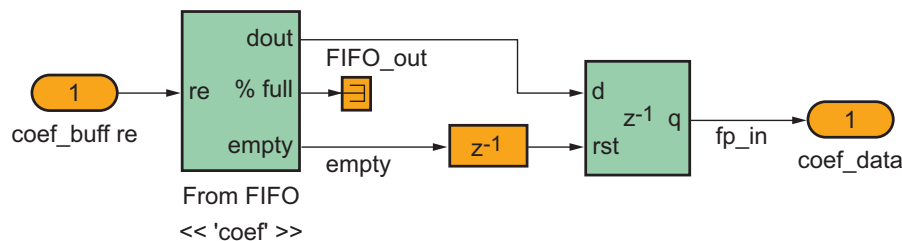


Figure 4 – System Generator-shared FIFO

data passing and function control. The FPGA coprocessor will appear as a hardware accelerator to the application software developer and is accessible through function calls.

The Xilinx Solution

Xilinx provides a complete DSP development environment for FPGAs based on The MathWorks’s Simulink and MATLAB modeling environments. Algorithms described in floating-point MATLAB can be synthesized into DSP functional blocks for Xilinx® FPGAs using AccelDSP. System Generator enables the use of Simulink for combining these blocks with a library of more than 90 Xilinx-optimized DSP blocks to form complete FPGA-based DSP systems.

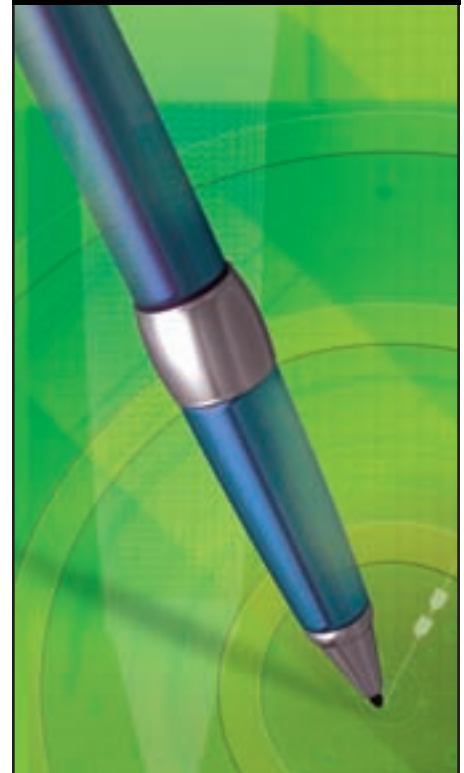
System Generator supports hardware co-verification, where part of the software simulation is replaced with implementations running on hardware. This allows you to validate your implementations in hardware

to one of the shared memories in the application software, defined in driver files automatically generated by System Generator. Interrupt generation is also supported in this flow to allow efficient execution between the processor and coprocessor.

Conclusion

The parallel processing power of FPGAs can significantly improve the performance, cost performance, and power dissipation in video, imaging, and telecommunications applications that benefit from parallel DSP processing, or those that require optimized, multi-channel processing. Heterogeneous, reconfigurable DSP platforms supported by a platform-based design methodology allow traditional DSP designers who are not familiar with FPGA design to quickly evaluate the benefits that an FPGA coprocessor can bring to their unique applications. ●●

GET PUBLISHED



WOULD YOU LIKE TO BE PUBLISHED IN XCELL PUBLICATIONS?

It's easier than you think!

Submit an article draft for our Web-based or printed Xcell Publications and we will assign an editor and a graphic artist to work with you to make your work look as good as possible.

For more information on this exciting and highly rewarding program, please contact:

Forrest Couch
 Publisher, Xcell Publications
xcell@xilinx.com

