WP212 (v1.0) March 18, 2004

DSP Co-Processing in FPGAs: Embedding High-Performance, Low-Cost DSP Functions

By: Steve Zack, Signal Processing Engineer
Suhel Dhanani, Senior Solutions Marketing Manager

FPGAs have been used in DSP applications for years as logic aggregators, bus bridges and peripherals. More recently FPGAs have been gaining considerable traction in high-performance digital signal processing applications and are emerging as ideal co-processors for standard DSP devices. In these latter two roles FPGAs provide tremendous computational throughput by using highly parallel architectures, and are hardware re-configurable; allowing the designer to develop customized architectures for ideal implementation of their algorithms.

The new generation of FPGAs developed using 90-nm process technology not only provide an effective way of implementing high-performance DSP functions but also provide the designer with an even more cost-effective solution. This white paper takes a look at some common high-performance DSP functions and calculates their effective implementation cost based on the published pricing of the underlying FPGA.

For the purposes of this calculation, we will look at the Xilinx Spartan-3™ FPGA family (introduced in 2003) which has a range of advanced features that enable the area-efficient implementation of DSP functions.

Using FPGAs for Implementing DSP Functions

In many cases, FPGAs work in conjunction with a conventional DSP – typically integrating pre- and post-processing functions, along with high performance signal processing. FPGAs can also integrate all the logic, bus-bridging, and peripheral functions, thus reducing system costs and affording a higher level of system integration.

FPGAs bring two key advantages to digital signal processing. First their architectures are well suited for highly parallel implementation of DSP functions, allowing for very high performance. Second, user programmability allows designers to trade-off device area vs. performance by selecting the appropriate level of parallelism to implement their functions. By programming the FPGA to use more on-chip resources, designers can achieve higher performance. By using less resources (and accepting a corresponding lower performance), designers can optimize the design for low cost.

These advantages are further illustrated below.

- FPGAs are essentially arrays of uncommitted logic and signal processing resources. These allow the designer to implement DSP functions using highly scalable, parallel processing techniques.

  For example, whereas a traditional DSP solution would implement multiple MAC functions in a serial manner, an FPGA allows designers to implement these in parallel using dedicated multipliers and registers that are now available in many FPGAs (illustrated in Figure 1).

  As an example of the real-life benefit of this capability consider a 256-tap FIR filter. By using resources that are available in the FPGA fabric, the designer can design a highly parallel implementation and achieve higher performance.

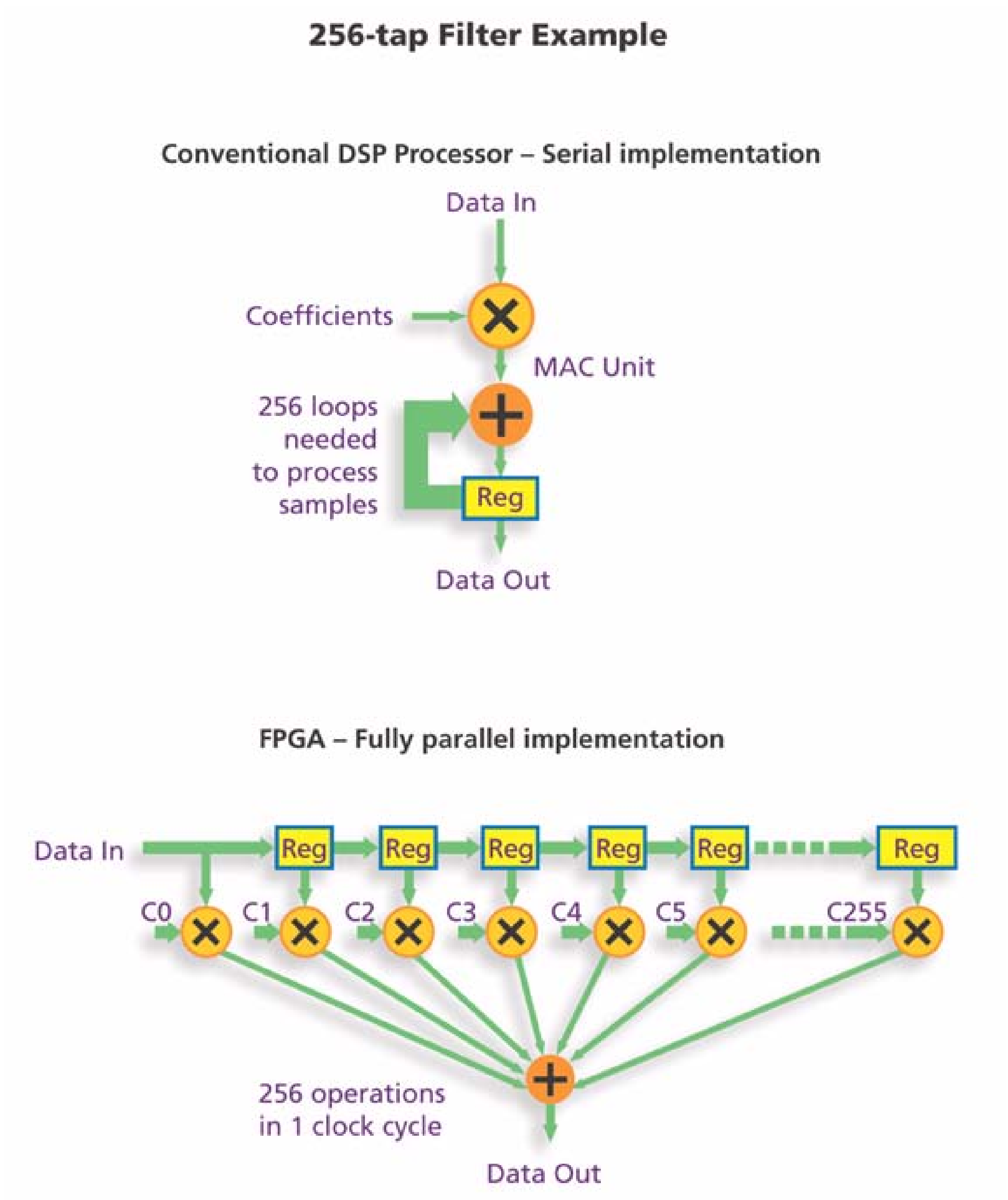
256-tap Filter Example
Conventional DSP Processor – Serial implementation
Data In
Coefficients
MAC Unit
256 loops needed to process samples
Reg
Data Out
FPGA – Fully parallel implementation
Data In
Reg
C0
C1
C2
C3
C4
C5
C255
256 operations in 1 clock cycle
Data Out

Figure 1: **FPGAs Parallel Approach to DSP Enables Higher Computational Throughput**

- Because FPGAs are completely hardware configurable, the designer has the flexibility to use only the necessary resources that the algorithm demands (see Figure 2). Put another way, the designer can optimize the hardware architecture to suit the ideal algorithm. In Figure 2 we show the different ways of implementing four multiply-accumulate (MAC) functions. By using four embedded multipliers within the FPGA fabric, this can be done at maximum speed. Alternatively the designer can opt to conserve area and implement the same function at a lower performance, by using only one multiplier, one accumulator, and a register; or the designer can use the semi-parallel approach shown in Figure 2.

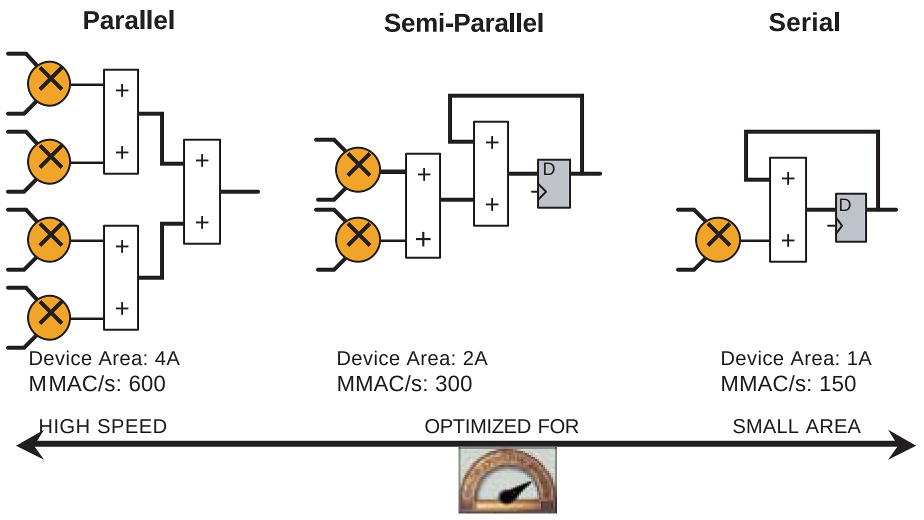

Figure 2: **Customize the FPGA to Suit Your Needs**

While FPGAs bring significant benefits to digital signal processing, it is important to analyze the effective cost of implementing DSP functions within the FPGA fabric. For the purpose of this analysis, the new Spartan-3 FPGA family is considered. This is the latest low-cost FPGA family from Xilinx that combines both low unit costs and system features for DSP.

Spartan-3 FPGAs – Architecture Optimized for DSP

Spartan-3 FPGAs use the latest 90-nm manufacturing technology to achieve low silicon die costs. These devices are also the only low-cost FPGAs that have all the features required for efficiently implementing DSP functions. Features that were once the exclusive domain of high-end FPGAs are now available within the Spartan-3 low-cost fabric – see Table 1.

Table 1: **List of Spartan-3 Features that Enable DSP Functions in an Area-Efficient Manner**

Spartan-3 Silicon Features	Customer Benefits
Embedded 18x18 Multipliers	Area efficient implementation of multiply-accumulate function
Distributed RAM	Local storage for DSP coefficients, small FIFOs
Shift Register Logic	16-bit Shift Register ideal for capturing high-speed or burst mode data and to store data in DSP applications
Up to 104 18 Kb Block RAM	Video line buffers, cache tag memory, scratch-pad memory, packet buffers, large FIFOs

With this family of FPGAs, the designer can implement high-performance complex DSP functions in a small fraction of the total device, leaving the rest of the device free to implement system logic or interfacing functions – providing both lower costs and higher system integration.

Table 2 demonstrates how the combination of advanced features and low cost work together to provide digital signal processing capability at a low cost. This table shows a sampling of available Spartan-3 parts, the number of embedded multipliers within each device, and the cost for MMAC/s using the 50,000 unit price for each part.

Table 2: **Calculating Cost Per MMAC/s**

Device	Embedded Mults (18x18)	MMAC/second (Number of Mults x 150 MHz)	Cost for MMAC/s[1]
XC3S50	4	600	$0.0055
XC3S200	12	1,800	$0.0024
XC3S400	16	2,400	$0.0030
XC3S1000	24	3,600	$0.0037
XC3S1500	32	4,800	$0.0044

Notes:

1. Calculated using customer resale price for 50,000 units – slowest speed grade and smallest package.

The "Million Multiply-Accumulate per second" (MMAC/s) column is calculated by multiplying the number of multipliers with the operating frequency of the multipliers, which in the case of the Spartan-3 FPGAs, is 150 MHz in the slowest speed grade.

Then looking at the published 50,000 price for the slow speed grade of the appropriate device, we calculate the cost for MMAC/s. This is one of the quoted industry benchmarks. With cost per MMAC/s approaching less than quarter of a cent – Spartan-3 FPGAs provide an economically viable DSP co-processing option.

Spartan-3 FPGAs – Achieving Lowest DSP Function Cost

There is no standard way of estimating the actual cost of implementing DSP functions onto FPGAs. For the purpose of this analysis, we introduce the notion of using effective cost which is determined as the cost based on percentage of the silicon area that is utilized, multiplied by the unit device cost. *There are various ways of estimating effective cost, but effective cost based on the percentage of the device utilized seems to be a fair way of calculating the cost to the user; since the remainder of the FPGA can be used to implement other system functions.*

To calculate the effective cost of a DSP function when implemented in an FPGA, we considered the Spartan-3 XC3S1000 device which is a mid-range member of the entire Spartan-3 FPGA family.

In many cases a given DSP function uses not only the FPGA logic but also embedded multipliers and block RAMs. In that case we estimate the amount of die space taken by these embedded functions and add that to the die area used by the logic.

Table 3 shows some of these functions and the cost of implementing these within the Spartan-3 silicon.

Table 3: **DSP Function Effective Costs in Spartan-3 Devices[1]**

Functions	% of the XC3S1000 Device Utilized	Effective Cost[2] (50K Units)	Key Specification	Other Specifications
1024-point complex FFT	24.1%	\$3.23	20 µs transform	20 µs transform, burst I/O, 16-bit input and phase factor
Single channel 64-tap FIR filter	3.0%	\$0.41	8.1 MSPS	16-bit data and co-efficient, MAC implementation, 8.1 MSPS
Digital down converter per channel	18.6%	\$2.49	Sample rate 100 MSPS	
Digital up converter per channel	18.6%	\$2.49	Sample rate 100 MSPS	
Viterbi decoder	37.8%	\$5.06	1.9 MSPS per channel	Parallel mode, trace-back 42, constraint length = 7, 32-channel, 1.9 MSPS per channel
Reed Solomon G.709 encoder	1.3%	\$0.17	120 MHz	
Reed Solomon G.709 decoder	6.9%	\$0.92	60 MHz	

Notes:
1. These costs do not include the cost for the programming PROM – since in many cases the existing EPROM on-board can be used for programming the FPGA. Effective costs are calculated using the 50,000 unit customer resale prices (slowest speed grade, smallest package).
2. Using Xilinx direct 50,000 unit price of \$13.40 for the XC3S1000 device.

Some of the most common functions used in DSP applications are Fast Fourier Transforms (FFTs) and FIR filters. A single channel 64-tap MAC FIR filter running at 8.1 MSPS can be implemented for an *effective cost of \$0.41*. Note that this filter utilizes 200 logic slices and four embedded multipliers – approximately 3% of the die area.

Forward error correction DSP cores such as Viterbi and Reed Solomon functions can be implemented at a low cost within the Spartan-3 device. A 32-channel, parallel mode Viterbi decoder running at 1.9 MSPS per channel has an *effective cost of \$5.06 or \$0.16 per channel*. A Reed Solomon G.709 decoder function running at 60 MHz takes only 6.9% of the same device *(effective cost \$0.92)*.

Complex functions such as a digital down converter (DDC) or a digital up converter (DUC) – commonly used in wireless base stations – take less than 20% of the XC3S1000 device *(effective cost of \$2.49)*.

Other Benefits

In addition to the inherent cost advantage that the Spartan-3 family offers, there are a host of other advantages to using these FPGAs in a DSP application:

- Ability to integrate system logic – Since the Spartan-3 architecture has all the features optimized for DSP functions, the implementation is extremely area-efficient. There are still plenty of resources left to handle traditional FPGA tasks such as the interfaces between other devices on the PC board. Both the system logic and the DSP functions can be implemented in one cost-effective fabric.
- Ability to integrate the system control path – Many systems have an external microcontroller. Xilinx offers a powerful 32-bit soft processor called the MicroBlaze™ processor that can be integrated within the Spartan-3 fabric. This processor which occupies only 6% of the XC3S1000 device (with performance up to 68 D-MIPS), can address many traditional 16 and 32-bit microprocessor applications within control, instrumentation, test and measurement, automotive, and high-end consumer applications.

Development Tool Flow

With Xilinx, DSP designs can be done using industry standard development tools. Using MATLAB® and Simulink® from MathWorks, coupled with Xilinx System Generator™ for DSP, system engineers can now model, simulate, and verify their signal processing algorithms, on their target hardware platform, all without leaving the Simulink environment.

The design flow typically involves the following steps:

1. A DSP designer develops and verifies the hardware model using industry-standard tools from MathWorks in conjunction with the Xilinx System Generator for DSP.
2. With a push of a button, Xilinx System Generator generates an HDL circuit representation that is bit- and cycle-true meaning that the behavior is guaranteed to match the functionality seen in the Simulink/System Generator model.
3. Then the ISE design tools synthesize the design and produce a bitstream that can be used to program the FPGA.

The error-prone and time-consuming step of having an FPGA designer translate the system engineer's design into HDL is thus eliminated.

With recent advances in the Xilinx System Generator, DSP designers can now generate an FPGA bitstream directly using Simulink/System Generator.

Figure 3 shows a typical design flow using the Xilinx System Generator.

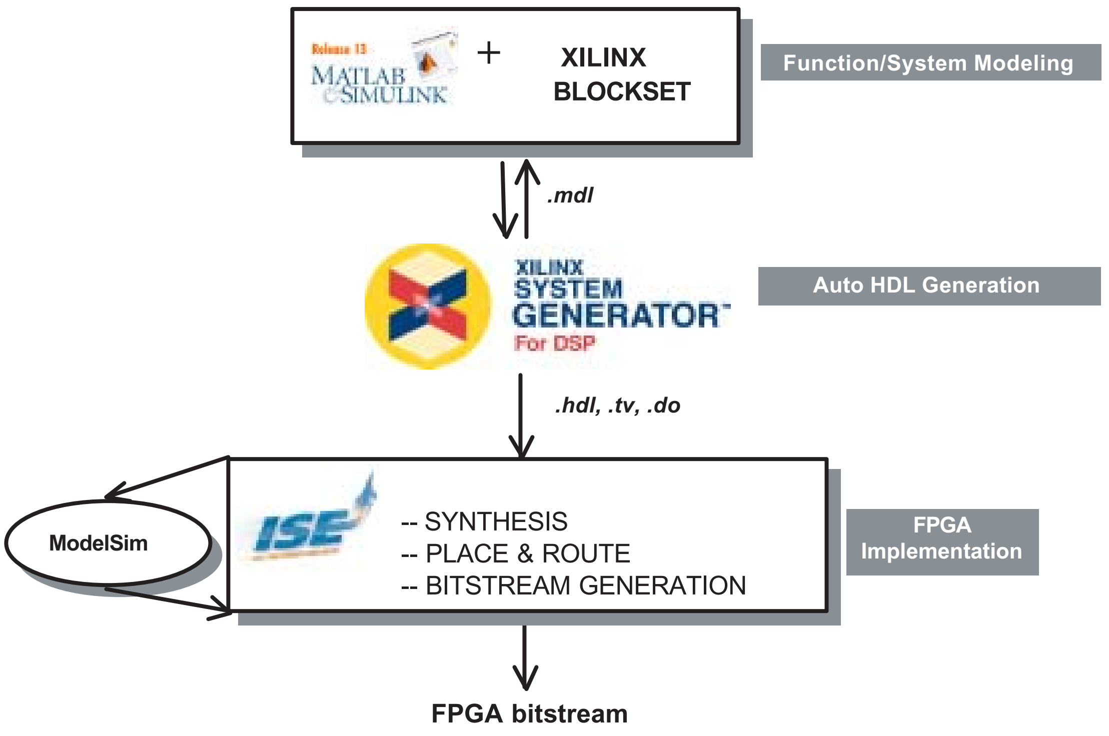

Figure 3: **DSP Design Methodology That Works Within an Existing Tool Flow**

Conclusion

With its combination of low unit costs and architecture optimized for DSP functions, Spartan-3 devices now enable the industry's lowest price-points for high-performance DSP functions. Xilinx further enables embedded DSP functions by providing design tools that fit within the standard DSP designer's tool flow, and enhances productivity by automating the FPGA implementation process.

With the availability of the Spartan-3 devices, the associated design tools, and the increasing number of off-the-shelf DSP functions optimized for this fabric, designers must evaluate embedding DSP functions within the Spartan-3 FPGA as one of the options when designing systems.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
03/18/04	1.0	Initial Xilinx release.