FPGA signal processing for radar/sonar applications

FPGAs have been used in radar and sonar systems for some time—mostly in support functions. However, as FPGAs get more sophisticated, performing and integrating more DSP functions in FPGAs is becoming the standard. As a result, radar- and sonar-system designers can rely on logic providers for optimized processing solutions, rather than on proprietary math functions.

By Ryan Kenny

Radar and sonar applications are signal-processing intensive and heavily rely on the efficient implementation of such digital signal-processing (DSP) algorithms as filtering, transforms and modulation. In past systems, conventional digital signal processors were used to perform many of these algorithms. However, field-programmable gate arrays (FPGAs) deliver an order of magnitude higher performance than traditional DSPs.

Changing requirements in radar applications

Modern military radar systems have evolving requirements, both in how the systems are designed and how the end user uses the data. This results in some of the same design changes in electronics systems affecting both the military and commercial design communities. That is, the need for smaller, energy-efficient systems with high processing-power requirements. This makes low power consumption a key driver in most designs.

With warfare having become more urban, ground clutter and background noise take on additional significance for the radar operator, thus demanding more processing power and better algorithms. Overlaying data from multiple sensors and known terrain features is one approach to increasing resolution, but this too has impacts on system-processing requirements and user-decision models.

An all-FPGA design for signal processing

Fitting multiple DSP functions into a single FPGA has many integration challenges, but also offers significant advantages to the designer in performance and flexibility.

The primary reasons for integrating DSP functions into a single FPGA are system-level reductions in size, weight and power. For example, eliminating the transfer path-
ways between separate FPGAs and DSPs significantly reduces power consumption and, therefore, heat. This, in turn, reduces the system-cooling burden of the design. Recent releases of design and place-and-route software, such as Altera’s Quartus II design suite, have advanced power-awareness features that significantly reduce dynamic power use of the FPGA. These options can be important to the designer; the benchmark of device logic density among competitive FPGA providers is beginning to give way to functionality-per-watt metrics, due to the sensitivity of power and cooling requirements in emerging systems.

Performance is also a key driver as FPGA-pipelined signal processing has become more reliable and faster than traditional processing technologies. In applications where performance is the driving parameter, efficiency can be sacrificed for application speed, where a memory-intensive, massively parallel floating-point math operation is desired. Alternatively, highly iterative DSP calculations can be implemented for applications where moderate performance is allowable, but where logic-element usage is limited.

This logically leads to the advantage of flexibility. The designer has the flexibility to decide between high-speed performance and the number of logic elements in every DSP operation, whereas calculation bandwidths and iterations would be more difficult and costly to modify in a dedicated DSP device. In addition, consolidating DSP functions within an FPGA allows for post-design system changes in the signal-processing architecture, whereas using separate DSPs locks the designer into a fixed set of chip interfaces once the board is designed. FPGA designers can alternately switch between 9-bit, 18-bit or 36-bit or 18-bit complex math functions without changing the system hardware.

Additional flexibility can be designed into the system when the designer uses fast-embedded processors for the execution or routing of complex floating-point operations. These functions are useful for radar applications.

**FPGA DSP functions in radar/sonar applications**

Several DSP functions are needed for radar or sonar processing near the receiver element. Each function should be closely examined to determine whether the application will show substantial speed and performance improvements through implementation in an FPGA. In some cases, these operations can be efficiently implemented using an FPGA embedded processor, even for highly complex and adaptive operations.

When a radar or sonar application calls for these operations to be performed with floating-point arithmetic, FPGAs have significant flexibility advantages if the design team takes advantage of a strong architecture-based design approach. Large floating-point math operations can be performed in standard logic cells (the least efficient option), in dedicated reduced-instruction-set-computer (RISC) embedded processors (the most flexible option), or in dedicated floating-point multiplier logic (the most efficient option).

FPGA providers and third-party developers offer efficient and accurate floating-point operators, Fourier-transform tools and filter compilers to FPGA designers as intellectual property (IP). Engineers should conduct their own research on the current availability of advanced DSP functions, but a great deal of preliminary information can be obtained through the technical representatives of programmable logic device (PLD) vendors.

**Digital up/digital down signal conversion**

The upconversion and downconversion of high-frequency signals are experiencing a dual migration, both into the digital domain, and into the same monolithic device (either the ASIC or FPGA) that performs the baseband processing. This push toward more digital, software-radio-style signal-processing techniques provides significant advantages to the system in signal accuracy and speed.

The closer to the RF front end (or the acoustical transceiver front end in sonar systems) that signals can be digitized, the fewer the analog-signal vulnerabilities that are introduced to the system. This includes high-order mixing products, error-vector-magnitude (EVM) impairments due to phase/magnitude imbalance, carrier feed-through, harmonics, and sideband noise. More important than signal integrity, however, is the design flexibility that the digital domain allows the radar-system designer. Dynamic filtering and conditional signal-processing algorithms significantly improve performance, as well as reduce implementation losses and the time required for the design cycle. While these advantages involve trade offs between power consumption and digital bandwidth, modern FPGAs provide designers much greater flexibility in mitigating power consumption, including the support of selectable core voltages, or critical-path power analyses.

The greater the numbers of on-chip resources available in FPGAs, the more designers are enabled to incorporate polyphase filtering and downconversion in the digital domain, as reflected in Figure 3. Multiple onboard or external numerically controlled
oscillators (NCOs) can allow very high phase discrimination with high-capacity FPGA devices. This application is useful for prototyping, research and development, where designers can incorporate and test multiple phase resolutions without significant hardware investments by using hardware-in-the-loop test methodologies.

**Algorithmic functions**

Examples of algorithmic math functions in radar systems include recursive least-square and square-root operations. Many designers have implemented these functions in C-based processors (in fixed-decimal and floating-point operations), or with proprietary FPGA VHDL operations. The current generation of FPGA devices include embedded processor and logic-cell resources to efficiently implement these processes; future generations will also have these capabilities. Additionally, IP cores and reference designs are becoming available to transition anywhere from dozens to hundreds of these operations into a single FPGA.

Tools are available to translate processor-based algorithms from C code to hardware languages, such as very high-level descriptive language (VHDL). These tools can be used to optimize certain logic functions from a standard main processor into an FPGA co-processor operating in parallel with the main processor, or to move entire operations from the main processor to the FPGA hardware. This provides an additional dimension of flexibility to the radar- or sonar-architecture designer’s toolkit.

**Complex matrix inversion**

Matrix inversion is an important element of adaptive-array designs and standard spatial-transceiver-array processing (STAP). These operations are commonly performed in fixed hardware elements, though efficiently implemented embedded processing has been demonstrated in some radar/sonar development programs.

The logic-element size and potential parallelism of a matrix inversion engine depends on the size of the array used in the radar system. As the size of the array is increased, so does the number of floating-point multiplications required by the system. Therefore, in larger arrays, there are more trade-off options between the speed of the system and the number of logic elements required by the system (both of which increase as the parallelization of the architecture increases).

Implementing this function using a combination of a DSP and a group of internal memory blocks is the most likely design path for radar-system designers. As these operations are often tailored to the adaptive-array algorithms of the radar system, they are likely to be custom designed in VHDL. However, reference designs that are optimized for the place-and-route capabilities of an FPGA device can be offered or designed-to-order from the FPGA manufacturer, if required for the radar or sonar system.

**Fast-Fourier transforms**

The bandwidths of many systems, including radar/sonar and test/measurement systems, are beginning to exceed the capabilities of dedicated DSPs. Implementing fast-Fourier transforms (FFTs) and their inverses in FPGA logic has advantages in prototyping and scalability, and offers design flexibility between a system’s speed and the number of required logic elements. For example, massively parallel implementations can be designed and distributed among the logic elements of a single or multiple FPGAs. However, while these implementations can significantly reduce latency, they impose the penalty of a greater number of logic elements.

In fact, the primary flexibility advantage of an FPGA for FFTs is the ability to select the optimal balance between these two parameters in the initial design. This is fortunate, because the implementation of large or complex FFTs should be the primary factor in any design, and the advantages of an FFT implementation in an FPGA (Figure 4) are apparent. However, creating code or modifying existing code from previous designs can be cumbersome when testing and verifying code units. Therefore, what is needed is a comprehensive suite of FFT design tools that allows a nearly infinitely scalable FFT design. These tools are available and they allow scripted logic distribution among multiple FPGAs where necessary. They can also automatically generate numerical coefficients having floating-point accuracy.

Because radar, sonar and digital-communication system designers must focus on the complications of multi-element beam-forming and waveform generation—not FFT design—programmable logic vendors such as Altera have internal tools and generators for conducting large, difficult element transformations. This includes reference designs and core IP wizards for standard and non-standard designs, as well as FFT co-processors, which are important design aids in the programmable logic offering.

**Design flow**

DSP logic designs are commonly executed from an initial model in simulation languages, such as Matlab or Simulink. These models are the most common, but not the only sources for designers to access optimized DSP IP offered through FPGA providers. The linkage between modeling and hardware implementation is important, not only for design simplicity, but for simulation and verification against the model.

As the design density for FPGA-based sensor systems increase, full system modeling and simulation will become more time consuming. Compile, simulation, and place-and-route times will increasingly become discriminators when selecting FPGA and design-software vendors. Furthermore, multiprocessor and distributed processing options for design software will be necessary to keep up with design complexity.

To cope with these trends, and to achieve the greatest signal-processing performance in their sonar or radar systems, designers are encouraged to consider options beyond their own VHDL modules or other internally developed IP. Specifically, they should consider working with programmable logic manufacturers to develop tailored DSP cores, or find ways to improve and optimize their designs through advanced place-and-route methods available for FPGA design tools. This is because the advanced capabilities of integrated circuits enabled by increasingly sophisticated fabrication technologies cannot be fully harnessed without flexible and effective design techniques.

**ABOUT THE AUTHOR**

Ryan Kenny is the technical marketing manager for military and aerospace products at Altera. He is responsible for creating FPGA-based technical solutions for military-data and signal-processing applications. He joined Altera in 2007, and has more than 10 years of experience in space and defense electronics in the U.S. Air Force and at Lockheed Martin.