FPGA TRADEOFFS FOR SOFTWARE RADIO APPLICATIONS

Rodger H. Hosking (Pentek, Inc., Upper Saddle River, NJ, USA, rodger@pentek.com)

ABSTRACT

In the last decade, ASICs (application specific integrated circuits) and DSPs (digital signal processors) have been deployed to handle nearly all signal processing functions associated with radio communications. However, FPGAs must be deployed judiciously with appropriate consideration for both the technical and business requirements of each project. This paper provides an overview of the signal processing requirements for software radio and a discussion of critical tradeoff issues involved in arriving at the best design strategy.

1. INTRODUCTION

Even though programmable logic has been around for decades, recently introduced FPGAs are now so powerful that they are now displacing both ASICs and DSPs in the latest software radio applications. In order to better understand how FPGAs can be used most effectively for software radio, we start with an analysis of the basic functions. We will then compare the suitability of FPGAs for these functions as compared first to ASICs and then to DSPs.

2. FPGAS FOR SOFTWARE RADIO FUNCTIONS

Figure 1 shows a typical wireless communication software radio system with various signal processing tasks identified. Usually, the digital down converter or digital receiver section is handled in a dedicated ASIC device consisting of three major blocks: the mixer, the local oscillator and the filter.

The local oscillator or NCO consists of a phase accumulator, which is just a register and an adder available as standard library blocks for virtually all FPGAs. The phase value in the accumulator drives a sine/cosine lookup table, which can be implemented as a simple ROM (read-only-memory). The mixer is nothing more than a pair of digital multipliers, now available as dedicated hardware resources in recent FPGAs. The decimating low pass filter usually consists of several CIC filter stages followed by an FIR filter and IP (intellectual property) cores for all of these basic functions are available from multiple sources.

Programmable DSPs are often used to implement some common demodulation, decoding and analysis functions. However, FPGA vendors and third parties now offer a good selection of IP libraries to handle Viterbi, Reed-Solomon, Turbo, convolutional and trellis decoders, DES engines, CDMA matched filters, and various noisy channel models.

To help with control functions, newer FPGAs now feature on-chip dedicated micro controllers supported with C programming development tools. Alternatively, IP cores are available for emulating popular legacy processors.

The benefit here is that by using a well-supported core processor you can take advantage of existing code and the software development tools already in place for these engines to help speed development and improve maintainability. For speech and video signals you can take advantage of a wide range of cores including ADPCM, JPEG and color space converters.

Several vendors are now offering FFT IP cores with scalable engines ranging in size from 16 to 16k points and higher using complex radix 4 algorithms. Since an FFT operates on blocks of data, the block memory RAM available with the newer FPGA devices allows some design tradeoffs. You can save memory by doing an in-place FFT with a single block, or move up to a swinging buffer arrangement to provide continuous real-time calculations. Some of the IP cores support several different memory models.

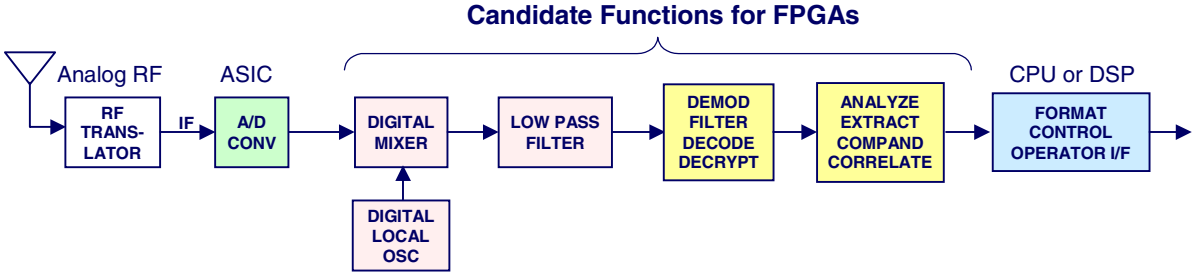

Figure 1. Typical Software Radio Receiver Block Diagram

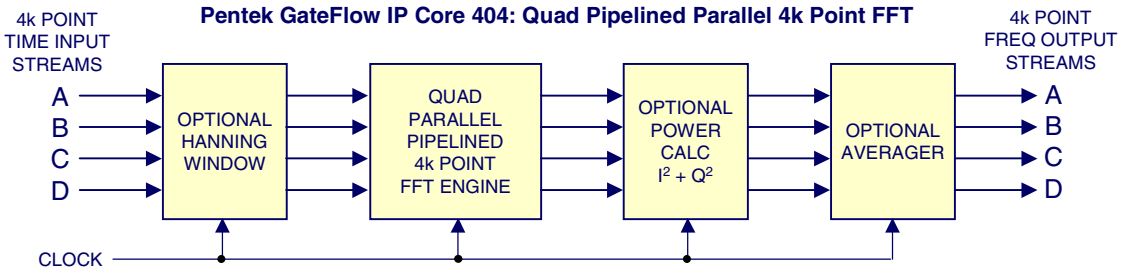

Figure 2. 4k Point FFT Intellectual Property (IP) Core for Xilinx FPGAs

One example of an IP core is the Pentek GateFlow Quad 4k FFT illustrated in Figure 2. Designed for the Xilinx Virtex-II family of FPGAs, it takes full advantage of hardware multipliers and distributed memory to implement a highly parallel architecture with pipelined stages to enhance speed.

For each clock cycle, it accepts four streams of time domain samples at the input and delivers four streams frequency samples at the output, fully reordered. Every 4096 clocks, it performs four 4k-point FFTs in parallel, resulting in an effective calculation time of 6.4 microseconds per FFT at a clock frequency of 160 MHz. Both real and complex data types are supported and optional input windowing, output power calculation, and averaging are available.

To put this performance level in perspective, a 500 MHz G4 PowerPC performs a 4k-point FFT in about 105 microseconds, or about 16 times slower than the FPGA. The obvious lesson here is that FPGAs should be seriously considered for these kinds of processing-intensive DSP algorithms.

The point is, if some or all of these signal processing functions are handled by an FPGA, the general purpose programmable DSP or CPU is now able to concentrate on the more complex and application-specific analysis and control functions, much more appropriate for its capabilities.

3. TRADEOFFS: ASICs vs. FPGAs

Although digital receiver functions can be implemented within an FPGA, it can draw more power and cost much more per channel than an ASIC, depending on many different factors like sampling frequency, filter characteristics, and signal-to-noise requirements.

Usually, the ASIC digital receiver has been designed with a full set of standard operating modes and features and has been more thoroughly tested and characterized than a custom combination of IP core building blocks available for FPGAs. This gap will obviously shrink as the cores become more complete. Since the ASIC hardware is optimized for dedicated functions, the latest ASIC devices are usually the better choice for extremely high-performance receiver applications.

The ASIC normally benefits from a longer user history, bug fixes and a much more complete characterization and testing effort. Standard ASICs usually have extra bells and whistles you may not need today but they have already been tested and will be ready to use for future requirements.

However, if a standard digital receiver ASIC is simply not available with just the right phase and frequency characteristics, or if the local oscillator just does not meet your switching requirements, the flexibility of the FPGA might be the appropriate recourse. Also, for proof-of-concept systems or when time-to-market is critical, FPGAs are often the right choice.

In other applications, where the required signal-to-noise ratios, filter skirts, or frequency templates are beyond the complexity of the standard filter inside a commercial ASIC, the flexibility of IP core filter designs for FPGAs can provide custom characteristics.

Another shortcoming of some digital receiver ASICs is the ability to provide output sampling that is decoupled from the input sampling rate and synchronous with the output symbol rate instead. Although, interpolation or re-sampling filters and synchronizers are now appearing on some new ASIC devices, they may not meet the needs of some of the new wireless protocols.

4. TRADEOFFS: DSPs vs. FPGAs

While FPGAs can handle many of the tasks traditionally perform on a programmable DSP chip, there are several factors worth considering.

Be careful of memory budgets for DSP applications. Even though today's FPGAs have generous on-board RAM, it may still a far cry from the large external SDRAMs normally surrounding a DSP chip. In spite of all the best design tools and simulators, DSP code (like any software) always seems to need more memory sooner or later. This usually occurs at the most inappropriate time in the design cycle - at the end. Newer FPGAs are now capable of embedding SDRAM controller cores to help alleviate this shortcoming.

While FPGA code can be reconfigured for new modes of operation and feature enhancements, it is usually much

easier to make the more significant changes on a programmable DSP instead. Sometimes, a very small change can have a profound impact on the gate and logic cell topology inside your device. In some cases, these changes can also mandate a new pin out requiring a new printed circuit board design.

5. SYSTEM EXAMPLE

As an example, Figure 3 shows a 32-channel digital receiver system suitable for a wide range applications including signal intelligence, direction finding, and signal tracking or dehopping receivers. It consists of a Model 6230 VIM-4 mezzanine module attached to a Model 4294 Quad G4 PowerPC VIM processor board at the bottom.

The receiver module has four 14-bit 80 MHz A/D converters for digitizing IF or HF analog inputs entering through front panel SMA connectors. All four A/Ds feed a bank of eight quad digital down converter ASICs with four channels of local oscillator, mixer and filter in each chip.

On board are two Xilinx Virtex-E FPGAs, each receiving the sixteen baseband signals from four quad receiver chips. These FPGAs are used to handle data formatting and channel selection for delivery down through the VIM interface to the 32-bit mezzanine FIFOs on the processor board. Since the receiver signals flow through the FPGAs, they can also be used to perform demodulation and decoding functions to offload these tasks from the DSP.

Note that all four A/Ds are connected to all eight quad receiver chips. Inside the front end of each receiver chip is a programmable crossbar switch that allows each of the four narrowband channels inside to independently select any one of the four A/D inputs.

With this arrangement all 32 receiver channels on board can independently select any of the four sources. This provides a very dynamic antenna-to-channel assignment scheme for systems that need to adapt to changing traffic patterns.

Also notice that the digital outputs of two A/D converters are delivered directly into each FPGA. This allows wideband A/D data to stream directly through the FPGA to the DSP, and the high bandwidth of the VIM interface supports clock rates up to 100 MHz.

Even more important, it also allows the FPGA to perform signal processing algorithms on the raw A/D data before it goes to the DSP, again, to offload some of its processing tasks. After accommodating the factory standard functions of data formatting and control, nearly 70 percent of these resources are still available for custom applications.

The factory default code takes care of all of the basic functions for most applications, including selection of channels for delivery to the DSP board, selection of real or complex data modes, selection of receiver data or raw A/D data, and selection of data packing modes. For custom signal processing tasks, an optional design kit for each product includes the VHDL source code for all of these standard factory default modes.

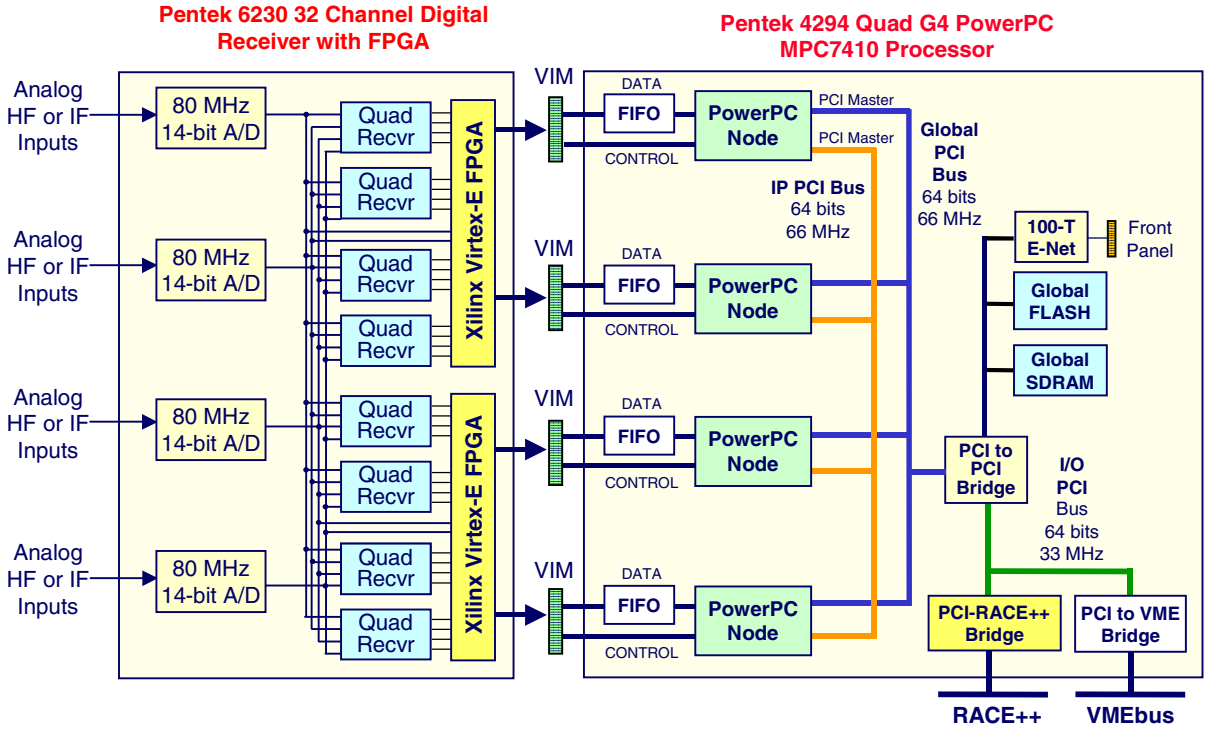

Figure 3. Complete 32-Channel Software Receiver and Signal Processing System for VMEbus

Customers can extend the factory code by adding their own algorithms at appropriate points in the signal flow path. FPGA algorithms can be developed and simulated by drawing on the wealth of IP cores and design tools available for these devices. Once compiled, the custom code can be downloaded through utility loaders into a non-volatile user memory on the board.

6. DESIGN GUIDELINES

Care should be taken when evaluating benchmarks for FPGA devices and the associated IP cores. Often they assume that data has already been loaded into internal block memory and that the operation is finished when the result is written back into internal memory. Getting the data into and out of these memories takes extra time and it must be taken into account.

Many IP cores are parameterizable, so that designers can specify the number of bits of calculation to tradeoff space for accuracy, if necessary. Be sure that the IP cores have enough precision to meet the system requirements.

Algorithms run faster when exception handling resources are omitted, and this can make benchmarks look deceptively fast. Designers may be able to guarantee through system architecture that certain exceptions simply cannot occur, but mysterious problems can appear in deployed systems if this aspect of the design is overlooked.

To help validate new designs, take full advantage of simulation tools. Many of the advanced simulators are now bit-true, meaning they perform the operations with exactly the same number of bits used in the FPGA hardware.

Be sure to allow enough time to thoroughly test a new FPGA design under all modes of operation to make sure it will act as reliably as the high-volume standard ASIC being replaced.

7. CONCLUSION

In spite of all of these caveats, FPGAs are very successfully taking up many of the roles formerly played by DSPs and ASICs. One of the major benefits of the IP core concept is that, like high-level languages, these cores can migrate to the next generation devices. This means that the wealth of core functions can accumulate and diversify, eliminating the need to start over again for each new family.

An excellent source for more information about devices and cores is the internet. New announcements are appearing daily and a good springboard for information is the third party section of FPGA vendors' web sites. Certain third parties are true experts in niche application areas and many of them offer consulting or custom design services as well.