Configurable DSP Technology for Wireless Communication Systems

Mazen A. R. Saghir

Department of Electrical and Computer Engineering
American University of Beirut
P.O. Box 11-0236 Riad El-Solh, Beirut 1107 2020, Lebanon
mazen@aub.edu.lb

Abstract - This paper provides an overview of configurable DSP technology and illustrates it applicability to wireless communication systems. It begins by tracing the evolution of configurable DSP architectures and discusses the tools used to customize and program these architectures. It then looks at software-defined radios to demonstrate how configurable DSPs can help enable new wireless applications.

I. Introduction

Digital signal processors (DSPs) are specialized devices designed to implement digital signal processing algorithms on streams of digitized signals. DSPs are widely used in wireless systems, where they are used to perform various filtering, encoding, decoding, and transform functions

The highly competitive nature of the wireless communications market and constantly evolving communication standards have resulted in short design cycles and product lifetimes. This environment has led to the emergence of a new class of configurable DSPs, which can leverage hardware flexibility, programmability, and reusability, to provide highly customizable DSP solutions.

This paper provides an overview of configurable DSP technology and shows how it can be used in wireless applications. Section II describes the different types of DSPs and how they are used in wireless systems. Section III then describes the evolution of configurable DSPs. Finally, Section IV describes how configurable DSP technology can be used to enable new wireless applications, such as software-defined radio.

II. DSPs in Wireless Systems

DSPs are widely used in wireless communication systems where they perform a variety of functions ranging from multiple-channel data processing in base stations to simple filtering in handsets. DSPs can broadly be divided into two classes: fixed-function application-specific integrated circuits (ASICs) and general-purpose programmable processors.

ASICs implement complex algorithms in hardware and are mainly used in applications that demand high computational performance. ASICs are widely used in wireless systems to implement computationally-intensive functions and complex algorithms, such as RF filtering, adaptive beam-forming and multi-user detection.

Programmable DSPs, on the other hand, are used to implement low- to medium-complexity algorithms in software, and are mainly used in applications that require good performance at low system costs. For this reason, programmable DSPs are widely used in wireless handsets to perform various base-band processing functions such as filtering, equalization, and echo cancellation.

As communication standards evolve and time-to-market pressures yield shorter design cycles, programmable DSPs become more appealing than ASICs due to the ease with which their functions can be modified. However, when hardware acceleration is the only approach for satisfying the computational demands of an application, ASICs maintain their advantage.

Configurable DSPs are a new class of DSP that aims to provide the best of both worlds: software programmability and hardware acceleration. The next section describes the evolution of configurable DSPs.

III. The Evolution of Configurable DSPs

A. Programmable DSPs

Programmable DSPs may be regarded as the earliest form of configurable DSPs, since their functions could be modified simply by changing the software they execute.

The first programmable DSPs appeared in the early 1980's and were based on a number of architectural features aimed at efficiently implementing key digital signal processing algorithms, such as FIR filters, IIR filters, and FFTs. These included fast multipliers for implementing single-cycle multiply-accumulate (MAC) operations; Harvard memory architectures and multiple on-chip buses for accessing instructions and multiple data operands simultaneously; tightly-encoded instructions that could specify limited instances of parallel operations; control structures for implementing low-overhead looping; and support for specialized addressing modes, such as modulo and bit-reversed addressing.

To maximize the utilization of these architectural features, early DSPs were mainly programmed in assembly languages. This made application development, debugging, and maintenance very tedious and error-prone. However, given their good performance, low system cost, and low power consumption, these DSPs have remained popular in applications such as wireless handsets, where cost is a major design constraint. Examples of this class of DSPs include the Analog Devices ADI 21xx, Lucent 16xx, and Texas Instruments TMS320C54x [1].

As programmable DSPs proliferated and applications became more complex, DSPs evolved to meet the increasing computational demands. Starting in the mid 1990's, new DSPs began to emerge with architectural features aimed at exploiting higher levels of instruction parallelism. For example, many DSPs, such as the Lucent 16xxx, Infinion Carmel, and DSP Group PalmDSP, featured dual MAC units. More recent DSPs, such as the Lucent StarCore, Analog Devices TigerSHARC, and Texas Instruments TMS320C6x include between two and four MAC units. In the case of the TMS320C6x, the data path is built around a Very Long Instruction Word (VLIW) architecture, which allows up to eight instructions to be executed in parallel. Some of these DSPs also provide single-instruction multiple data (SIMD) instructions that perform parallel operations on groups of packed data [1].

Although such DSPs provide high levels of performance, they are still too expensive and power-hungry to be used in handheld or battery-powered devices. That is why they are commonly used in communication infrastructure equipment, such as data routers or wireless base stations, where they typically replace older systems that would have required multiple DSP boards with less powerful DSPs to perform similar functions. For example, a single TMS320C6416 DSP operating at 600 MHz can process 300 adaptive multi-rate voice channels at 12.2 kbps or 35 data channels at 384 kbps [2].

Unlike earlier DSPs, newer DSPs benefit from better programming tools, such as optimizing high-level language compilers. Although most DSPs today are programmed in languages like C, assembly programming is still used to optimize performance-critical portions of an application or to enhance memory use.

In addition to providing higher levels of parallelism, some recent programmable DSPs provide specialized hardware units or instructions for accelerating specific digital signal processing functions. The hardware units are typically programmable, providing designers with a high degree of flexibility to match the specific requirements of the intended target application. For example, the Motorola MSC8101 combines a StarCore DSP with an enhanced filter coprocessor (EFCOP) [3]. The EFCOP is fully-programmable, and can be used to implement real and complex FIR, IIR, adaptive FIR, and multi-channel FIR filters. This makes the MSC8101 particularly well suited for applications like channel equalization and echo cancellation. Similarly, the Texas Instruments TMS320C6416 includes a programmable Viterbi coprocessor and Turbo coprocessor for decoding convolutional codes and turbo codes, respectively [4][5]. Finally, the Analog Devices ADSP TS101S TigerSHARC [6] includes special chip-rate- and symbol-rate-optimized instructions for performing such operations as complex MAC; add/compare/select, which is useful for accelerating trellis functions in Viterbi and Turbo decoders; and enhanced turbo decoding for accelerating MAC*log mapping functions.

The main advantage of these types of DSPs is the ability to leverage their hardware resources and instruction sets to accelerate specific applications. However, such specialization also makes these DSPs less useful in applications where the available hardware resources or instructions cannot be well utilized.

B. Field Programmable Gate Arrays

For high-end digital signal processing where the highest possible performance is needed at the lowest cost and power consumption, ASICs are still the processors of choice. ASICs are widely used in wireless systems for implementing chip-rate functions such as RF filtering, spreading/despreading, adaptive beam-forming and multi-user detection. However, ASICs require very long design and development times and are very expensive to design and manufacture. Moreover, ASICs are inherently rigid and are not very well suited to applications that are constantly evolving. For these reasons, field programmable gate arrays (FPGAs) have emerged as an alternative to ASICs in wireless communication systems.

FPGAs consist of arrays of logic and interconnection blocks. The logic blocks consist of SRAM-based look-up tables that can be programmed to implement arbitrary logic functions. Once programmed, the logic blocks can be interconnected using a programmable connection fabric to form hardware designs of varying complexity. FPGAs are programmed using special CAD tools that map designs developed in hardware description languages, such as Verilog or VHDL, to the underlying logic and interconnection blocks. This mapping is done over several steps that include logic synthesis, technology mapping, placement, and routing [7].

FPGAs are mainly used for the flexibility they provide. Like programmable DSPs, FPGAs are programmed and configured in software. This makes it very easy to upgrade or add functionality to an FPGA, even if it is already deployed in the field. Since most DSP engineers may not be familiar with programming FPGAs, vendors like Xilinx and Altera provide intellectual property (IP) block libraries for common DSP functions [8][9]. These blocks, which include Viterbi, Turbo, and Reed-Solomon coders and decoders; various filters; and FFTs; are typically parameterizable and can be customized

through GUI-based tools for specific applications.

Like ASICs, FPGAs achieve high levels of performance by implementing complex algorithms in hardware. FPGAs are particularly well suited for accelerating algorithms that exhibit a high degree of data-flow parallelism. In fact, it is not uncommon for FPGAs to execute algorithms like FIR and IIR filters 10 to 100 times faster than programmable DSPs [10]. Although FPGAs are still too expensive and power-hungry to be used in wireless handsets, they are nonetheless widely used in infrastructure equipment like data routers and wireless base stations.

C. Configurable DSP Cores

Continued advances in VLSI fabrication technology and increasing transistor densities have ushered in the era of the system-on-a-chip (SoC). Today, it is possible to design entire systems, including processors, memories, buses, and interfacing logic on a single silicon chip. By integrating all components on the same chip, execution performance is improved and overall system costs are reduced significantly.

SoCs are built using commodity intellectual property (IP) components. These are pre-designed and pre-tested macro cells that are commonly available as licensable hard or soft IP cores from specialized semiconductor manufacturers. Hard IP cores are technology-dependent components that have a fixed layout, while soft IP cores are technology-independent components described in a synthesizable hardware description language, such as Verilog or VHDL. Among the common types of IP cores used in SoCs are programmable DSPs from vendors like DSP Group or LSI Logic [1]. These cores may be integrated with other IP cores to perform various control or data processing functions. For example, in a cell phone, a microprocessor core may be used to process user commands and coordinate various phone functions, while a DSP core is used to compress, decompress, and filter speech signals. The combination microprocessor-DSP is commonly referred to as a computational platform, and is used mainly to balance the data processing workload.

Traditionally, programmable DSP cores have been designed around fixed data paths and instruction set architectures. However, with the proliferation of hardware description languages and computer-aided design (CAD) tools, companies like ARC, Tensilica, Chameleon, and Improv Systems now offer soft processor cores whose data paths and instruction set architectures can be customized by system designers to meet specific design constraints. Using proprietary CAD tools, designers can configure the number and type of functional units to include in a data path, the size of a register file, the width of the address and data buses, and the amount of instruction-level parallelism that can be supported by the data path. More importantly, designers can define new instructions to accelerate critical sections of target application software. Such user-defined instructions are executed on special hardware accelerator blocks that either are a fixed part of the processor architecture or that are custom designed for a specific function. Once a processor has been configured, designers can use simulation tools to assess the impact of their design on performance and area. The design can then be fine tuned and optimized to meet the various design constraints. Once a design meets its performance, area, and power objectives it can be synthesized into a net list and implemented in an FPGA or a SoC ASIC. The result is a highly-optimized computational engine aimed at a specific application.

Although most configurable DSP cores are currently configured at design time, new research into configurable processor technology will soon make it possible to reconfigure cores dynamically, in much the same way an FPGA can be configured. For example, researchers at Northwestern University and the University of Washington have proposed integrating a reconfigurable functional unit (RFU) with a superscalar data path [11]. The RFU is based on an FPGA-like reconfigurable array, and is used to implement custom instructions that enhance the execution performance of application programs. The custom instructions are automatically generated by an optimizing C compiler using a number of heuristic algorithms. Researchers at the University of California at Berkeley have taken a different approach by using a reconfigurable array as a co-processor for a MIPS processor [12]. The reconfigurable array is used to accelerate performance-critical loop nests, and a C compiler is used to map the loop nests on the configurable array automatically. Finally, researchers at Hewlett-Packard Labs in Palo Alto, California have developed a design framework to automate the generation of custom embedded processors [13]. The processors are based on an EPIC architecture [14] to exploit available levels of fine-grain parallelism. They also include a non-programmable accelerator (NPA) block that is used to implement performance-critical loop nests. An optimizing C compiler is used to map the loop nests onto the NPA. Unlike the other research projects, where only the reconfigurable arrays can be changed, the HP researchers have developed a number of tools to explore the architectural design space automatically and generate pareto-optimal designs that meet specific design constraints. This again results in embedded processor cores that are highly optimized for specific applications.

IV. Configurable DSPs for Software Defined Radios

Although there still is no standard definition of software-defined radio (SDR), it is generally agreed that the concept involves a software implementation of user terminal functions to enable the terminal to dynamically adapt to the radio environment in which it is located. SDR also enables wireless operators to introduce new services independent of the wireless standards used. It also enables subscribers to benefit from new or customized services and truly global connectivity [15].

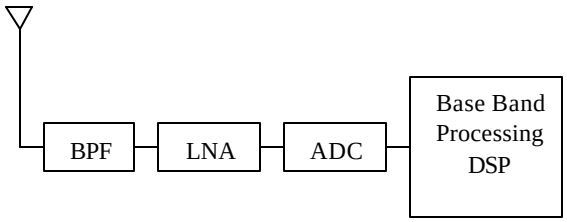

Figure 1: An ideal software radio receiver.

Figure 1 shows an ideal software radio receiver. The receiver has a simplified RF stage where the only analog components are the antenna, band pass filter (BPF), and low-noise amplifier (LNA) [15]. Wideband A/D converters are used to convert signals at the RF stage to enable the processing of IF and BB signals in the digital domain. This makes it possible to implement various transceiver functions in software and reprogram the transceiver when protocols change or evolve.

Currently, software defined radios are facing several technological challenges that make them difficult to implement. For example, jitter effects make it very difficult to perform A/D conversion in the RF stage. Moreover, a single RF stage for a multi-band system is not feasible due to the difficulties associated with building antennas and LNAs with very wide bandwidths ranging from hundreds of megahertz to units or tens of gigahertz [15].

On the processor side, programmable DSPs, with their fixed data paths and instruction-set architectures, are still not powerful enough to process many IF functions. FPGAs also are still too expensive and power-hungry to be used in wireless handsets. That is why upcoming configurable DSP cores, with their support for software programmability and reconfigurable hardware units, are particularly well suited for implementing SDRs.

Once the technological hurdles facing SDRs are passed, it is conceivable that future handsets will be built around SoCs that contain one or more configurable DSP cores and large memories for storing hardware configuration data and software implementations of various IF and BB functions. Reconfiguring and reprogramming such devices would be as easy as loading data from a smart card or downloading data over an air interface.

V. Summary and Conclusions

Configurable DSPs are a new class of DSP that combines software programmability with hardware flexibility. Configurable DSPs can be used to implement highly adaptable computational platforms that are optimized for specific applications.

Given the constantly evolving nature of wireless standards and applications, configurable DSPs are particularly well suited for implementing wireless systems and enabling new applications like software defined radio.

V1. References

[1] *Buyer's Guide to DSP Processors*, Berkeley Design Technology, Inc. (BDTI), 2001.

[2] N. Cravotta, "DSP Family Adds Members," *EDN Magazine*, p. 22, March 29, 2001.

[3] *MSC8101 User's Guide*, Motorola, www.motorola.com

[4] "Using TMS320C6416 Coprocessors: Viterbi Coprocessor (VCP)", *Texas Instruments Application Report SPRA750A*, August, 2002.

[5] "Using TMS320C6416 Coprocessors: Turbo Coprocessor (TCP)", *Texas Instruments Application Report SPRA749*, June, 2001.

[6] *ADSP-TS101S TigerSHARC DSP Product Brief*, Analog Devices, www.analog.com.

[7] S. Brown, R. J. Francis, J. Rose, and Z. Vranesic, *Field Programmable Gate Arrays*, Kluwer Academic Publishers, 1992.

[8] J. Hwang, B. Milne, N. Shirazi, and J. Stroomer, "System Level Tools for DSP in FPGAs," *Proceedings of FPL2001*, pp. 534-543, 2001.

[9] Altera, www.altera.com.

[10] J. Eyre, "FPGA/DSP Blend Tackles Telecom Apps," *Electronic Engineering Times*, July, 2001.

[11] Z. A. Ye, et. al., "CHIMAERA: High-Performance Architecture with a Tightly-Coupled Reconfigurable Functional Unit," *Proceedings of the 27th International Symposium on Computer Architecture*, pp. 225-235, 2000.

[12] T. J. Callahan, "The Garp Architecture and C Compiler," *Computer*, pp. 62-69, April 2000.

[13] V. Kathail, et. al., "PICO: Automatically Designing Custom Computers," *Computer*, pp. 39-47, September, 2002.

[14] M. S. Schlansker and B. R. Rau, "EPIC: Explicitly Parallel Instruction Computing," *Computer*, pp. 37-45, February, 2000.

[15] E. Buracchini, "The Software Radio Concept," *IEEE Communications Magazine*, pp. 138-143, September 2000.