

Bridging Dream and Reality: Programmable Baseband Processors for Software-Defined Radio

Dake Liu, Coresonic AB and Linköping University

Anders Nilsson and Eric Tell, Coresonic AB

Di Wu and Johan Eilert, Linköping University

ABSTRACT

A programmable radio baseband signal processor is one of the essential enablers of software-defined radio. As wireless standards evolve, the processing power needed for baseband processing increases dramatically and the underlying hardware needs to cope with various standards or even simultaneously maintaining several radio links. Meanwhile, the maximum power consumption allowed by mobile terminals is still strictly limited. These challenges require both system and architecture level innovations. This article introduces a design methodology for radio baseband processors discussing the challenges and solutions of radio baseband signal processing. The LeoCore architecture is presented here as an example of a baseband processor design aimed at reducing power and silicon cost while maintaining sufficient flexibility.

INTRODUCTION

OVERVIEW

The concept of software-defined radio (SDR) has been a dream over the last decades. In spite of a plethora of publications and prototypes, low-cost and low-power commercial SDR systems are not available in the volume market. One of the major gaps between the ideal SDR and the reality (traditional application-specific integrated circuits [ASICs]) is the baseband processing capacity. A novel Very-Large-Scale Integration (VLSI) implementation with both efficiency and flexibility will be the bridge that connects dream and reality.

A digital radio communication system converts a digital bitstream to radio frequency signals by digital modulation. The modulated signal, which is subjected to various interference, noise, and impairments during its propagation through fading channels, is then recovered by equalization and error correction coding in the receiver.

As the radio frequency spectrum is a scarce resource, researchers and engineers are trying to

improve the spectral efficiency (bits per second per Hertz) of the modulation methods used in modern standards. By improving spectral efficiency, more users can be allocated to a certain piece of spectrum.

Recently, new technologies such as orthogonal frequency-division multiplexing (OFDM) and multiple-input multiple-output (MIMO) have been used to increase spectral efficiency. Meanwhile, all these new technologies add a significant amount of complexity to baseband processing, which in turn presents a big challenge to the underlying hardware systems, demanding new computing solutions.

In addition to supporting advanced modulation schemes, several fundamental challenges of radio wave propagation must be managed.

One of the greatest challenges in wideband radio links is the problem of multipath propagation and intersymbol interference. Multipath propagation occurs when there are more than one propagation path from the transmitter to the receiver. Since all the delayed multipath signal components add up in the receiver, intersymbol interference is created. As the phases of the received signals depend on the environment, some frequencies add constructively and some destructively, thus destroying the original signal. Unless the transmitter and receiver sit within an echo-free space or other artificial environment, the transmission is usually subjected to multipath propagation. There is only one common radio communication channel that is usually considered echo-free: a satellite link. To mitigate effects caused by multipath wave propagation and high mobility, advanced equalizers are needed that require very high computing throughput.

Other types of interference such as burst noise and white noise must also be mitigated. Burst noise, also called glitch noise, consists of high-amplitude short pulses added to the received signal during radio transmission. The radio channel is usually filled with different kinds of burst noise from lightning and the switching of electric equipment. Since burst

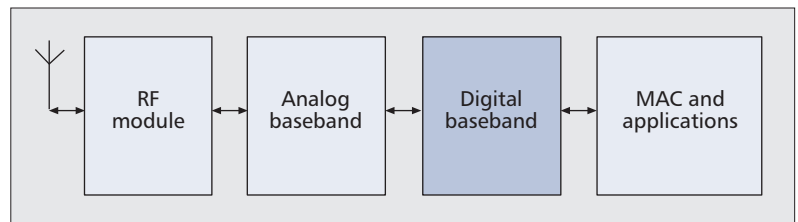
noise can be several magnitudes stronger than the useful signals, it destroys some of the transmitted signals. The way to eliminate the impact of burst noise is to change the burst noise to pseudo white noise and correct burst noise induced errors by normal error correction algorithms against white noise. Interleaving algorithms are used to spread out neighboring bits in a long bitstream. Interleaving is, in principle, data permutation. Interleaving (de-interleaving) requires significant amounts of memory and computing for irregular address generations.

Finally, channel coding is used to correct errors that occur during transmission. Forward error correction (FEC) algorithms (e.g., convolutional codes, turbo, and LDPC) require substantial logic and memory cost, and also add significant latency at the receiver side.

In the following, synchronization and various symbol processing tasks (channel estimation and symbol equalization) are briefly addressed.

Synchronization — In order to demodulate received signals into bits correctly, the receiver must be accurately synchronized to the transmitter radio frequency and the sampling clock of the transmitter. Carrier frequency offset (CFO) and sampling frequency offset will always exist since, in reality, there is a difference between the local oscillator frequencies of the RF transmitter and receivers as well as a sampling frequency offset — that is, a difference between the sampling frequencies of the analog-to-digital converter (ADC) in the receiver and the digital-to-analog converter (DAC) in the transmitter. In order to correctly extract the transmitted frame from the received signals, synchronization is needed at the receiver side. Modern wireless systems, especially those based on OFDM, are very sensitive to CFO and sampling frequency offset. As control messages for synchronization are usually extracted from correlation related operations, computing cost is not a problem as long as the relative distance is fixed between the transmitter and the receiver. However, when the mobile terminal is moving, a Doppler frequency directly proportional to the relative velocity is added to the ADC sampling frequency. This causes the CFO and sampling frequency offset to vary quickly. In such a case, intensive computation is needed to carry out synchronization more frequently.

Channel Estimation — The availability of accurate channel state information (CSI) is critical for the channel equalization process and hence for the overall performance of wireless systems. The estimation of CSI forms one of the most intensive tasks in radio baseband signal processing. Moreover, when the mobility of a handheld device increases, the coherence time of the channel (the lifetime of a radio channel model) will decrease. For example, the coherence time of a radio channel between a base station and a mobile terminal will be less than 5 ms when the relative velocity between them is more than 150 km/h, which is the normal speed of a train. This means that the CSI at the terminal side has to be updated once every 5 ms, which demands heavy computation.



■ **Figure 1.** A radio transceiver.

Equalization — After CSI is obtained through channel estimation, equalization is needed to compensate for channel fading and interference so that the transmitted symbol can be detected and de-mapped into log-likelihood ratio information which can be further exploited by the FEC decoders to decode the transmitted information bits. Especially as MIMO technologies have been adopted by more wireless standards, different equalization schemes ranging from linear detection (e.g., zero-forcing) to more advanced lattice-based detection [5] become popular. Each new proposal introduces higher decoding quality and heavier computational cost at the same time. Generally speaking, there is always a trade-off between performance and complexity due to the fact that baseband processing has hard deadline constraints.

CHALLENGES

As the complexity of wireless systems keeps increasing, reliable reception of a bit from a noisy channel requires thousands of arithmetic operations and memory accesses with complicated addressing algorithms — both at the receiver and transmitter. Therefore, a radio baseband processor is an essential and most challenging component in any advanced radio transceiver. In Fig. 1 the gray block is the baseband processor in a radio transceiver. In this figure MAC stands for media access control.

The data rate of mobile communication increases about 100 times every 10 years, according to Edholm's law. Meanwhile, the scarcity of radio spectrum prohibits further expansion of the radio frequency bandwidth in the long run. The only opportunity left to maintain this rule of thumb is to increase the spectral efficiency (e.g., to transmit more bits per second per unit of bandwidth). At the same time, more functions and different connections must be handled by the mobile terminal. A modern mobile terminal (e.g., smart phone or high-end laptop) should handle different links with different versions such as WiMAX (IEEE802.16e-2005), Third Generation Partnership Project Long Term Evolution (3GPP LTE), Wi-Fi (IEEE 802.11a/g/b/n), High Speed Packet Access (HSPA), Global System for Mobile Communications (GSM), Enhanced Data Rates for GSM Evolution (EDGE), wideband code-division multiple access (WCDMA), and Bluetooth. Moreover, the terminal should be able to function as a GPS receiver as well as a broadcasting receiver (DVB-T/H, ISDB-T, DMB-T, and DAB).

Each connection needs a radio baseband processor to recover data from the noisy and time variant channel under various interferences. The

Based on carefully specified function coverage, the goal of an ASIP design is to draw up an instruction set architecture for the specific application domain and to reach the highest performance over silicon, power consumption and design cost.

baseband computing load is usually very high, up to thousands of operations per bit when receiving data under high mobility from a long distance. For example, according to our experiences, to receive 10 Mb/s from a multi-user WiMAX channel, more than 30 Gop/s are needed in a mobile device. However, in order to achieve sufficient battery lifetime, a baseband module in a handheld device has strict constraints on power consumption (~100 mW). Therefore, to minimize power consumption, most traditional radio baseband modules have been implemented as ASIC modules to minimize overheads by giving up programmability.

However, a traditional solution without programmability faces three challenges. The first challenge is that the standard may be volatile and continue to evolve in the future. A change could result in redesign of an ASIC module. Redesign time is usually up to a year. The second challenge is to design a multimode mobile phone or laptop connection. If all radio links and broadcasting standards listed above are to be integrated into one mobile terminal, up to 16 baseband ASIC modules have to be included, which would result in heavy silicon and design costs and prohibitive cost. The third challenge is the quickly increased tapeout cost of advanced silicon technologies.

Using programmable hardware and software controlled hardware multiplexing not only maintains the silicon cost at a reasonable level, but also supplies more flexibility and prolongs the product lifetime. Programmable radio baseband processors are therefore essential for current and future SDR enabled mobile terminals. A programmable radio baseband processor can reach high silicon and power efficiency if the overhead is properly controlled using the application-specific instruction set processor (ASIP) design methodology [1]. Silicon efficiency is defined as the performance over silicon ratio. Power efficiency is defined as the performance over power consumption ratio. Examples show that a programmable radio baseband processor can consume less silicon area than an ASIC radio modem because a programmable solution can utilize the hardware resources by hardware multiplexing [2, 3]. The complexity of hardware multiplexing can easily be handled by running ASIP software. Handling the hardware multiplexing in ASICs, however, is more difficult. Designers seldom design radio baseband modules with hardware multiplexing (e.g., reusing one datapath for both transmitter and receiver) in an ASIC.

PROGRAMMABLE BASEBAND PROCESSOR DESIGN

DESIGN METHODOLOGY

A programmable radio baseband processor is an ASIP for radio baseband signal processing [1]. The design of an ASIP is different from the design of a general microprocessor. Designers of general-purpose processors consider ultimate performance and ultimate flexibility. The instruction set must be general to support unknown applications. ASIP designers must consider the

application and cost first. The function coverage shall be sufficient for the group of applications rather than ultimate. Usually, the biggest challenge for ASIP designers is the efficiency issue.

Based on carefully specified function coverage, the goal of an ASIP design is to draw up an instruction set architecture for the specific application domain and to reach the highest performance over silicon, power consumption, and design cost. In this article the scope of the application is to cover all digital radio baseband algorithms. The goal of the design is to handle all processing of baseband signals under the scope with satisfied data precision to reach sufficiently low bit error rates and required performance to meet the computing latency requirements of the physical layer specifications.

An application-specific instruction set will be designed based on code analysis techniques, and the instruction set will be evaluated based on DSP benchmarking techniques for radio baseband. In this case the benchmarking checks the performance of complex data computing (fast Fourier transform [FFT], convolution, frequency domain complex data processing), conflict-free parallel memory access, and parallel data manipulations for error correction.

EXPLORING ARCHITECTURES

All physical layer standards mentioned earlier in this article can be classified into:

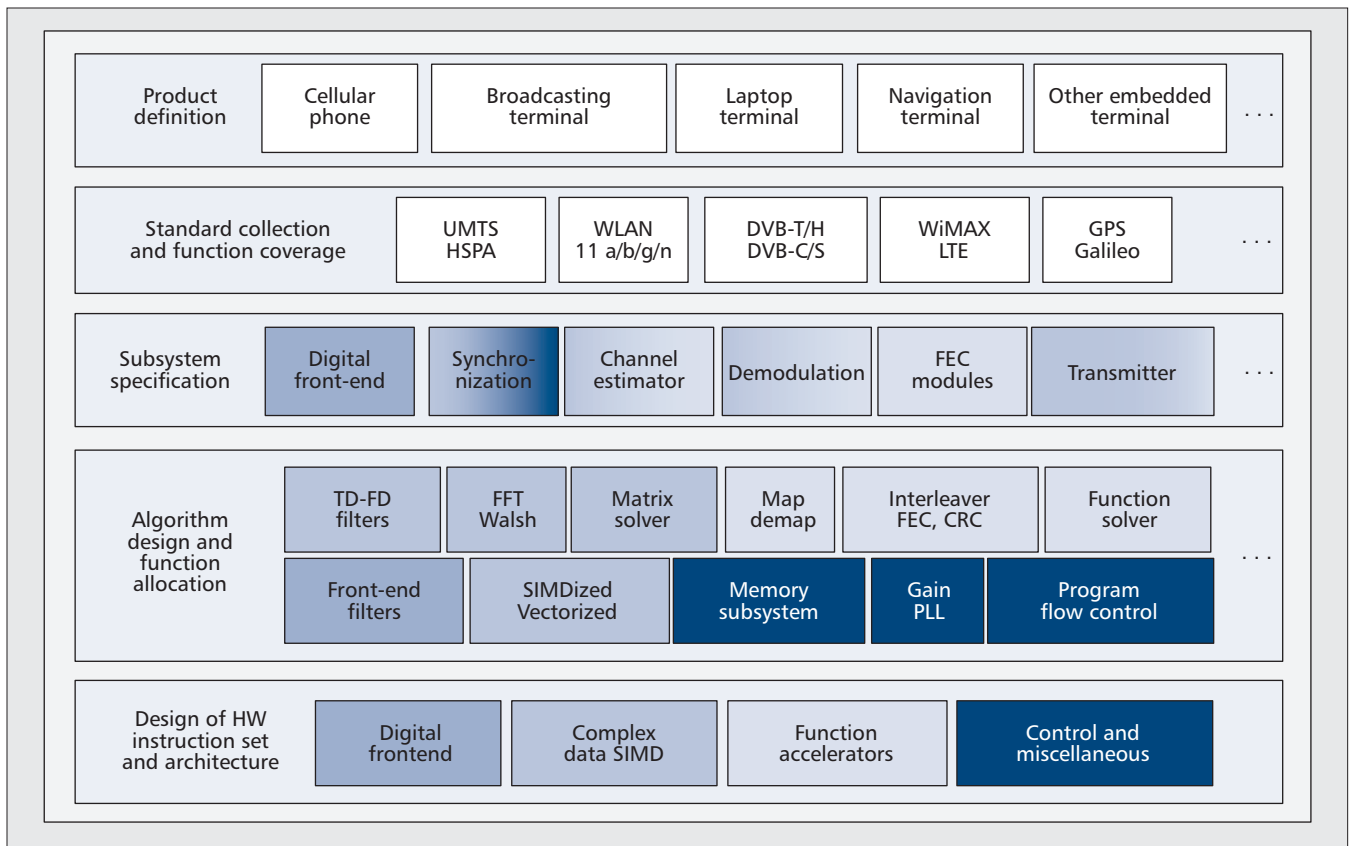
Traditional narrowband: Traditional narrowband systems such as GSM and Bluetooth-1 usually use a single carrier. When the bandwidth of a single-carrier radio transceiver is low, non-coherent detection algorithms are usually used for data reception, and the implementation of receivers is easy.

CDMA: This is a special kind of single-carrier radio transceiver. Typical CDMA systems are WCDMA (HSPA), CDMA 2000, TD-SCDMA, and IEEE802.11b. Multiple users share frequency and time resources by assigning different orthogonal spreading codes to different users to allow multiple accesses. When the bandwidth of a single-carrier radio transceiver is high, multipath reception with high mobility is a big challenge. The implementation of a multipath receiver is computing cost extensive.

OFDM/OFDMA: The orthogonal frequency-division multiplexing/multiple access (OFDM/OFDMA) system is the most widely adopted technology for current and future broadband data transmission and broadcast systems. Standards based on OFDM/OFDMA include IMT-Advanced, 3GPP LTE, WiMAX (IEEE 802.16e 2005), WLAN (IEEE 802.11a/g/n), DVB-T/H, and DAB.

MIMO: Recently, multi-antenna technologies have been proven effective in increasing the spectral efficiency of wireless systems; nevertheless, they also add a significant amount of complexity to baseband processing. In general, multi-antenna schemes fall into three categories: transmitter diversity, spatial multiplexing, and beamforming. Standards that have MIMO as mandatory or optional features include IMT-Advanced, HSPA, LTE, WiMAX, and WLAN (IEEE 802.11n).

The first design step of a programmable radio



■ **Figure 2.** Architectural exploration of a radio baseband processor.

baseband processor is to specify the product specification of the processor. A programmable radio baseband processor platform shall be used in cellular phones, laptop terminals, broadcasting terminals, global positioning systems, and embedded systems (Fig. 2). Following the product specification, related standards should be collected, and function coverage specified following these standards. Subsystems, such as front-end filters, synchronizers, channel estimators, channel equalizers, and error correction modules, shall be specified accordingly. Algorithms required by these subsystems will finally be specified and allocated to hardware. Basic algorithms are:

- Integer data filters and matching filters are mostly used for integer data filtering and correlations.
- Complex data filters and matching filters are used for low-pass and bandpass filters, sampling rate adaptation, identifying preambles, and correlation for synchronization of in-phase and quadrature-phase data.
- Transform algorithms include FFT, discrete cosine transform (DCT), and Walsh transform.
- Frequency domain signal processing includes frequency domain filters, pilot sub-carrier processing, channel estimation, and equalization.
- Division, square root of integer and complex data, waveform generator, and other most used computing extensive algorithms using lookup tables (LUTs) and function solver accelerators.

- Matrix computing in both frequency and time domains include matrix addition, multiplication, transpose, LU decomposition, QR decomposition, and singular value decomposition.
- Bit manipulations related algorithms, such as FEC and cyclic redundant check (CRC). Various FEC decoders such as convolutional, turbo, Reed-Solomon, and LDPC have been widely adopted in modern wireless systems.

Function coverage and performance requirements are obtained by application analysis and used to design the assembly instruction set. Here, the application is specified as the physical layer signal processing of radio transceivers for the listed standards. The definition of sufficient function coverage is given by the scope and the lifetime of a product. Analysis of applications shows that approximately 90 percent of the execution time is used to execute the seven listed classes of essential algorithms. This means that the system will be optimized if these seven algorithms can be executed in the most efficient way. This follows the so-called 90-10 percent code locality rule, which says that 90 percent of the execution time will be spent executing 10 percent of the code. Following the 90-10 percent code locality rule, an optimized architecture can be identified to accelerate the 10 percent of the code that consumes 90 percent of the execution time.

Following the 90-10 percent code locality rule, the radio baseband signal processor consists of four computing clusters, as shown in Fig.

To minimize the silicon and power overhead induced by program memory and instruction decoding, the instruction set must be customized only for baseband signal processing, with high efficiency yet sufficient flexibility.

2 in four shades: light medium blue, deep medium blue, light blue, and deep blue. Figure 2 shows the regularity of the radio baseband processor architecture. A radio baseband processor in any category (single carrier, CDMA, OFDM, and MIMO) can be further partitioned into four computing domains as four data path clusters or four processor cores. Each cluster or processor is one of the four heterogeneous computing domains:

Digital front-end, light-medium blue: A digital front-end (DFE) engine is the preprocessor of radio signals from an ADC. It consists of a cascaded integrator-comb filter (CIC) for decimation and low-pass filtering; a correlator for packet detection and waking up the baseband processors that follow; fractional sample rate conversion via interpolation and decimation (e.g., a Farrow filter); and symbol shaping for modulated signals. Since the DFE usually handles simple functions without significant variation, programmability is not mandatory. Instead, most DFEs are implemented as parameterized ASIC blocks to minimize the control overhead.

Complex data SIMD, a symbol processor, deep medium blue: A symbol processor handles all complicated computing for complex-data in time and frequency domains including complex data rotation, filtering, pattern matching, and matrix computing. It also performs transformations between time and frequency domains. To maximize the utilization of the symbol processor for all pipelined-parallel and massive-parallel computing as well as irregular complex computing, sufficient programmability for parallel complex-data SIMD signal processing must be supported by a comprehensive instruction set [2].

Function accelerators, light blue: It supports all functions between mapping / de-mapping and the interface toward the media access control (MAC) layer. It shall also support acceleration for function solving such as $1/x$, square root, and waveform generators. Because functions are relatively stable, FEC modules are usually not programmable and configurability is sufficient to support varies of algorithms in multiple standards.

Processor for control and miscellaneous functions, deep blue: This is usually a small microcontroller unit (MCU). The control processor handles miscellaneous functions such as phase locked loop control, gain control, DC cancellation control, program flow control, and task threading of multiple program flows. It is also responsible for configuration of hardware modules and system-level interconnections.

INSTRUCTION SET SPECIFICATION

To minimize the silicon and power overhead induced by program memory and instruction decoding, the instruction set must be customized only for baseband signal processing, with high efficiency yet sufficient flexibility [4]. Instruction efficiency is defined as the number of micro operations (arithmetic, logic, addressing, data moving) carried by an instruction. When an instruction becomes very efficient, the instruction will be strong for special algorithms, and its flexibility will be low. For example, the instruction FFT N is one instruction for N -step butter-

fly computing. It is very efficient and cannot be used for other algorithms. To achieve both high performance and sufficient flexibility, we need to have both accelerated instructions (e.g., a butterfly of an FFT, convolution, and vector operation) and reduced instruction set computer (RISC) instructions (e.g., a single step of arithmetic operation or simple data moving).

The coding efficiency of an instruction is measured by the number of micro operations over the length of the binary machine code. A most inefficient instruction uses uncompressed control signals as the machine binary code. The next inefficient instruction is the so-called very long instruction word (VLIW). The instruction holds the flexibility to control each part of the parallel data path, and memory accesses. A VLIW is essential when a processor is designed for unknown applications. However, for baseband application, all essential functions are collected in Fig. 2. It is not necessary to offer flexibilities beyond the coverage of the applications. Custom coding of instructions therefore promote coding efficiency.

A single instruction multiple tasks (SIMT) architecture combines task-level instruction, single-instruction multiple data (SIMD) vector instructions, and RISC instructions in one instruction set. It offers both the performance of task level and SIMD vector computing and sufficient RISC control flexibility at the same time [2]. The RISC instructions are used for multi-program flow control, data quality control, miscellaneous functions, and hardware/software configurations. Because radio baseband algorithms and their execution behavior are relatively well known during processor design, an SIMT instruction set is sufficiently flexible and the most efficient instruction set for radio baseband signal processing.

To efficiently control and use the hardware resources of the architecture given in Fig. 2, a SIMT instruction set consists of the following subsets of instructions:

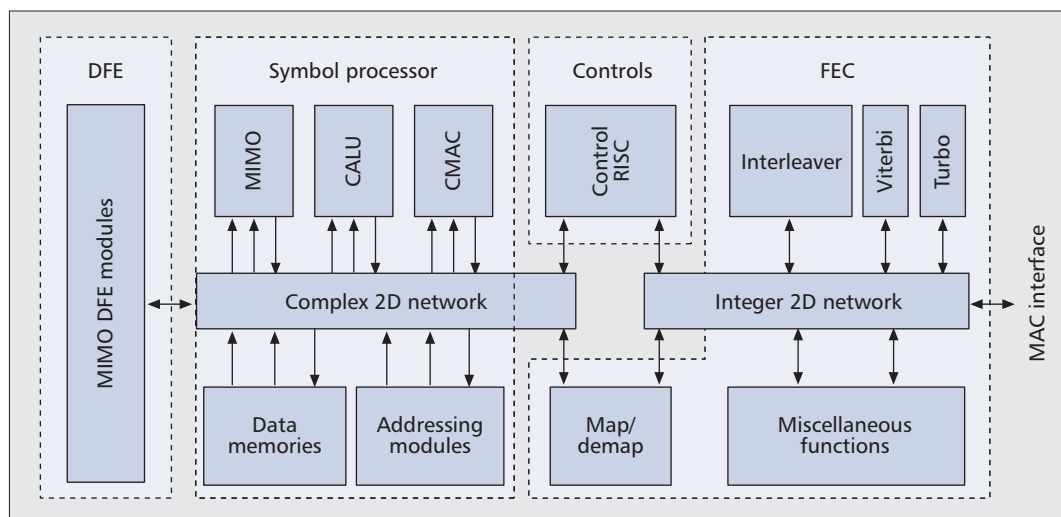
Task-level SIMT instruction subset: The subset contains vector instructions for parallel computing of complex vector data. A typical example is FFT $2 N M_x M_y M_c$. It executes radix-2 FFT butterfly N times using vector data from M_x , sending the result to M_y , and getting a coefficient from M_c . Similar instructions include N -tap convolution, N -points vector scaling, and vector data move.

Single-step SIMD instruction subset: An instruction contains a single-step computing of complex data, such as complex data multiplication or addition of (conjugated) complex data.

RISC instruction for miscellaneous functions: The subset contains normal RISC instructions and normal DSP instructions such as normal integer iterative computing instructions, normal load store instructions, and normal arithmetic instructions.

Special accelerated instructions: The subset contains special instructions for specific modulations. A typical instruction is for scrambling. MIMO instructions should also be specified, for example, the $1/x$ computing for complex data. Special instructions are also specified for fast data access based on special addressing algo-

The LeoCore baseband processor is presented here as a successful example of programmable baseband processor design based on the SIMT architecture. Both area and power figures of LeoCore outperforms previously presented non-programmable solutions.



■ Figure 4. LeoCore architecture.

Measured power consumption for the highest data rate of 31.67 Mb/s in DVB-T/H is 70 mW at 70 MHz. One special feature of a programmable solution should be mentioned here. The clock rate and power consumption of LeoCore can vary depending on selected algorithms. For example, by optimizing pilot processing algorithms, the clock frequency of DVB-T could vary from more than 100 MHz down to 70 MHz.

CONCLUSION

SDR is a promising technology to allow pervasive and versatile wireless connections. Programmable baseband processors are the key components enabling SDR. Based on the ASIP design methodology [1] presented in this article, by optimization through system and architecture levels, it is possible to develop a programmable radio baseband signal processor that consumes low power and low silicon area, while achieving sufficient performance and flexibility by correctly specifying the functional coverage and instruction set architecture. The LeoCore baseband processor is presented here as a successful example of programmable baseband processor design based on the SIMT architecture. Both area and power figures of LeoCore outperform previously presented non-programmable solutions [3].

ACKNOWLEDGMENTS

The work has been partially financed by SSF, the Swedish Foundation for Strategic Research, and the European Commission FP7 Multi-Base project.

REFERENCES

- [1] D. Liu, *Embedded DSP Processor Design, Application Specific Instruction Set Processors*, Morgan Kaufmann, 2008.
- [2] A. Nilsson, E. Tell, and D. Liu, "An 11 mm², 70 mW Fully Programmable Baseband Processor for Mobile WiMAX and DVB-T/H in 0.12μm CMOS," *Proc. ISSCC*, San Francisco, CA, Feb. 2008.
- [3] L.-F. Chen et al., "A 1.8V 250mW COFDM Baseband Receiver for DVB-T/H Applications," *ISSCC Dig. Tech. Papers*, 2006, pp 1002–11.

- [4] D. Liu and E. Tell, "Low Power Baseband Processors for Communications," Ch. 23, *Low Power Electronics Design*, C. Piguet, Ed., CRC Press, 2005.
- [5] D. Wu et al., "Implementation Aspects of Fixed-Complexity Soft-Output MIMO Detection," *Proc. IEEE VTC-Spring*, 2009.

BIOGRAPHIES

DAKE LIU [SM] (dake@isy.liu.se) is a professor and the director of Computer Engineering at Linköping University, Sweden, from which he received his Ph.D. in 1995. His research interests include ASIPs and on-chip multicore integrations, SDR platforms for future mobile terminals and radio base stations, and low-power electronics. He was a cofounder, and CTO of FreehandDSP Ltd. He is a cofounder and CTO of Coresonic Ltd. He was a senior ASIC designer in Ericsson Sweden from 1995 to 1998. He is currently a professor of embedded systems in Nanjing University, China. He worked for Beijing Jiaotong University, China, 1982–1990.

ANDERS NILSSON is principal system architect and co-founder of Coresonic Ltd. He received his M.S. in applied physics and electrical engineering and his Ph.D. in computer engineering from Linköping University, Sweden. His research interests include high-speed mobile wireless links, radio technology, and baseband processor design. He has three U.S. patent applications and is the co-author of *Radio Design in Nanometer Technologies* and *Handbook of WiMAX*.

DI WU [S'06] received M.Sc. degrees in electrical engineering from Beijing University of Posts and Telecoms, China and Linköping University, Sweden, in 2003 and 2005, respectively. Since February 2005 he has been with the Department of Electrical Engineering, Linköping University, working toward his Ph.D. His current research interests include software-defined radio, specifically multi-antenna systems, and their flexible VLSI implementations.

JOHAN EILERT received his M.Sc. in computer science from Linköping University, Sweden, in 2003. He joined the Division of Computer Engineering of the Department of Electrical Engineering in Linköping in 2003 as a Ph.D. student. His research interests include ASIP architectures and SDR, and custom floating-point datapaths.

ERIC TELL received his M.Sc. in applied physics electrical engineering in 2001 and his Ph.D. in computer engineering in 2005, both from Linköping University, Sweden. He is a co-founder of Coresonic and currently holds the position of principal designer at Coresonic. His professional interests are ASIP architectures, algorithms, and design methodologies for flexible baseband processing.