RECONFIGURABLE ACCELERATORS ENABLING EFFICIENT SDR FOR LOW-COST CONSUMER DEVICES

Geoffrey Burns (Silicon Hive, Eindhoven, The Netherlands; g.burns@philips.com); Paul Gruijters (paul.gruijters@philips.com); Jos Huisken (jos.huisken@philips.com); and Antoine van Wel (antoine.van.wel@philips.com)

ABSTRACT

Cost pressures present formidable barriers to widespread adoption of software-defined radios in consumer markets. System-on-chip platforms implementing radio base-bands for consumer applications typically rely on ASIC accelerators with limited or no programmability for compute-intensive processing DSP tasks, as are typical in IF or base-band (de)modulation of a digital radio. Porting such tasks to processors, general-purpose DSP's, or reconfigurable logic would overwhelm system development budgets in cost-constrained applications. Aside from silicon cost, any solution proposed to introduce efficient software programmability must also address the overhead and risk incurred by design teams in adopting unfamiliar technologies. A solution based upon programmable domain-specific accelerators, assembled on standard microcontroller platforms, is presented in this paper. Aspects of the accelerator architecture, programming methodology, and software library design are overviewed, along with an evaluation for application in the digital radio broadcast domain.

1. INTRODUCTION

The consumer domain contains numerous examples of products with significant demand for software programmability of features, along with simultaneous and contradictory cost and power constraints. Digital radio receivers provide one such example. In this market, radio receiver product designers attempt to introduce products in the face of multiple broadcast media, multiple transmission formats, and even evolving broadcast formats in some cases. As limited by cost constraints in this consumer market, practical programmable platforms have not been available to allow significant modification or introduction of several important radio receivers in the post-fabricated IC. Baseband processors for the emerging satellite digital audio radio services (SDARS), provide one important example of high datarate requirements that constrain implementation on programmable platforms. Similar examples of this cost problem occur in physical layer processors in other consumer domains, including cellular communications and wireless networks. Although demand for software programmability exists in these domains, the application is restricted due to the difficulty in realizing cost-efficient implementations using conventional programmable technologies.

Figure 1, illustrates a simplified view of current signal processing implementation technologies, organized in terms of computation efficiency (MOPS/Watt) per process technology node. The relationship in terms of MOPS/area is similar for many DSP applications. Reliable conventional technologies, with well developed programming flows, are available for applications with low computational requirements, or relaxed constraints on computational efficiency. In consumer applications demanding high computational efficiency, such as physical layer or baseband processors, the drive to minimize cost justifies the choice to forfeit flexibility and undergo expensive ASIC design projects.

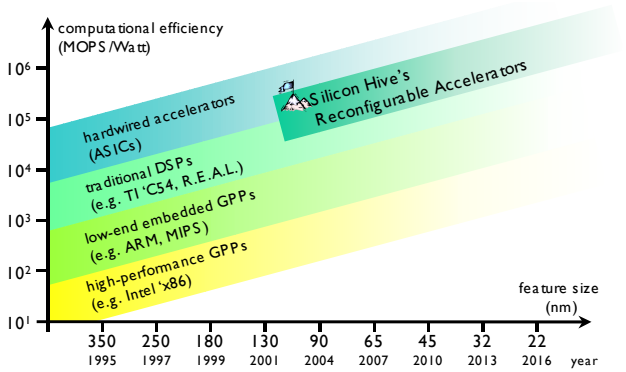

Figure 1: Computation efficiency of signal processing implementation technologies, and their progression with Silicon process technology. In cost-constrained applications which drive the selection of ASICs over conventional programmable implementation technologies, domain-specific programmable accelerators are used to provide a flexible implementation at computational efficiencies comparable to ASICs.

This paper provides an overview Silicon Hive's coarse-grained reconfigurable accelerator technology, which proposes to fill the gap illustrated in Figure 1, by

introducing a solution with computational efficiency comparable to ASICs. The technology comprises a family of domain-specific accelerators, along with a powerful c-based programming flow. Each accelerator can be integrated with familiar RISC/microcontroller platforms, assuming the same role intended for hardwired ASIC acclelerators in these platforms. As illustrated in Figure 2, the hardware/software partitioning used in standard platforms is a useful starting point for systems utilizing Silicon Hive reconfigurable processors. Within the new flexible platform, signal processing algorithms partitioned to ASIC accelerators in the standard RISC/DSP system-on-chip, are instead programmed onto a reconfigurable accelerator. In addition to flexibility, improved silicon efficiency can be achieved as multiple tasks are time-multiplexed on the same accelerator.

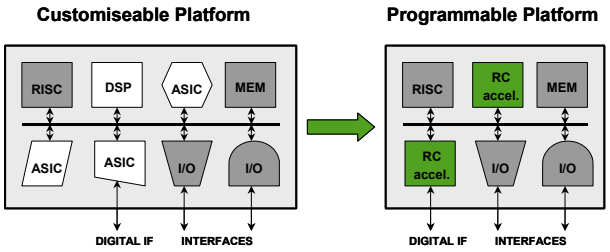

Figure 2: A standard platform consisting of a RISC, DSP, and ASIC accelerators offers limited customizability. This platform becomes fully programmable, yet remains practical, when the ASIC accelerators and DSP are replaced with domain-specific reconfigurable accelerators.

In the subsequent sections the technology will be explained and evaluated for the system-on-chip for multi-standard digital radio. First, the definition, architecture, and fabrication of the required reconfigurable accelerators are explained. The means to program and evaluate signal processing tasks onto a reconfigurable accelerator is then reviewed. Finally, silicon area and power for a multi-standard digital radio are evaluated.

2. PROCESSOR DEFINITION & DESIGN

2.1. Processor and Tool Generation

Silicon Hive's proprietary processor and tool generation methodology enables the convenient specification of specialized domain specific processors. Each accelerator is a specific instantiation from a general architecture template underlying the processor generation methodology. The architecture of the template embodies a coarse-grained reconfigurable computing strategy within a general parameterizeable structure. The structure of this template also facilitates automatic generation of processors and corresponding development tools from a concise machine description. Further detail on the architecture template and generation methodology is presented in [1][2]. After examining the requirements of the application, we can then define the processors and generate their synthesizeable description in VHDL, as well as the compiler, assembler, and cycle-accurate simulator.

2.2. Programmable Platform Requirements

In order to envision the processors required to build a multi-standard digital radio, we examine the block diagram in Figure 3 below. The block diagram broadly represents the functionality required to receive transmissions from either of the SDARS in the US, as well as terrestrial broadcasting systems including Eureka Digital Audio Broadcast (DAB) [4][5][7]. The maximum configuration is defined by the requirement to simultaneously demodulate one or more satellite transmissions, in addition to an OFDM transmission in SDARS [4][5].

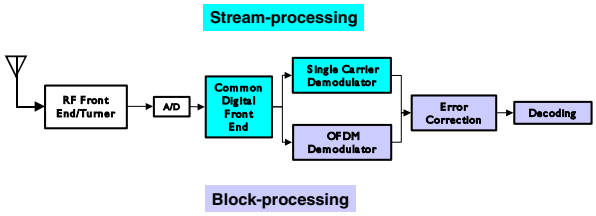

Figure 3: Simplified block diagram for a multi-standard digital radio.

The minimum set of accelerators required to construct the programmable platform of Figure 1 are determined by the signal processing organization and dataflow in the applications. For the multi-standard digital radio, we have processing organized around data framed in memory (block processing), and typically sample-based processing (stream processing) in the digital-IF, baseband satellite receiver, and time-domain processor of the OFDM demodulator. Avoiding the overheads we would introduce when using a single general-purpose architecture for implementation, we instead define two accelerators, each embodying a careful tradeoff between flexibility and efficiency for either block processing or stream processing.

2.3. Avispa Block Accelerator

The Avispa block accelerator, illustrated in Figure 4, has been developed for efficient and reconfigurable acceleration of DSP transforms, and other algorithms that may be organized around data samples framed in memory. Such processing includes the FFT, Frequency domain processing

for OFDM, Error Correction, and Source decoder in Figure 3.

The Avispa accelerator contains a highly parallel ULIW datapath, composed of five interconnected Processing and Storage Elements (PSE). The PSE, a basic component of the Silicon Hive architecture template, is a very-long-instruction-word (VLIW) datapath containing several issue slots with associated function units, register files, and local memory storage. An expanded view of one PSE is illustrated in Figure 4 below. The key attributes of the processor include:

- 4 16x16 multipliers
- 5 ALUs
- 4 1KB and 1 8KB local storage memories
- 4 4-butterfly per cycle add-compare-select units

The processor is capable of up executing up to 60 operations in parallel. Bus interface support includes the AMBA interface for easy integration with RISC-based hardware systems.

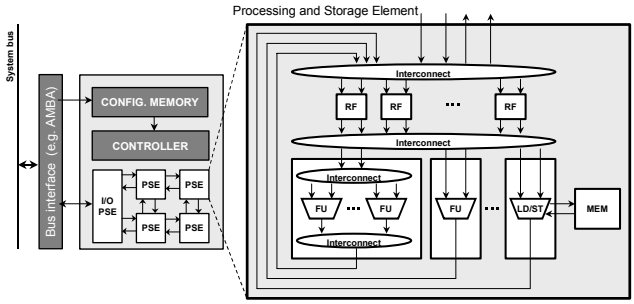

Figure 4: Architecture of the Avispa accelerator, a ULIW processor for flexible implementation of signal processing kernels framed in memory.

A layout of the Avispa accelerator is illustrated in Figure 5. The 115K NAND2 equivalent gates of logic to form the ULIW, along with the data and control memory, occupy approximately 6mm^2 when fabricated in a 0.13µ CMOS ASIC. A 2K point FFT running continuously on this processor will execute 31 operations in parallel, and dissipate 0.8mW/MHz of power.

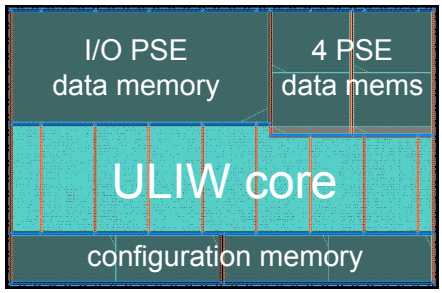

Figure 5: Layout of the Avispa accelerator in 0.13µ CMOS technology. The silicon area of this implementation is approximately 6mm^2, with the ULIW core composed of 115K NAND2 equivalent gates.

2.4. Avispa Programming Flow

A powerful suite of programming tools makes effective use of the massively parallel resources of the AVISPA ULIW, while abstracting the architecture details away from the programmer. The Hivecc spatial compiler reads kernel programs, such as an FFT or Viterbi decoder, each written in a restricted subset of ANSI-c. Advanced constraint analysis and scheduling techniques extract the parallelism within each algorithm kernel and generate an efficient schedule for each processor. In many cases, schedules are obtained where the majority of the algorithm is executed within a single-cycle DSP loop, a configuration desireable from the standpoint of both instruction size and power consumption [1].

Compiled schedules for each signal processing kernel can be pre-loaded in the processor configuration memory, signal processing chains to be constructed from function calls to these kernels. A cycle-true simulator is used to develop and evaluate signal processing kernels and chains. Consistent with the Silicon Hive processor generation methodology, the compiler and simulator are generated in conjunction with the processor, according to the original processor specification.

2.5. Bresca Stream Accelerator

The complement of the Avispa accelerator is the Bresca streaming accelerator, specialized for efficient software-defined implementation of algorithms with extreme and dynamic data-rates, as encountered in the IF or base-band front-end of a (de)modulator. As illustrated in Figure 6, Bresca is an array of single-PSE TIM-generated processors, each capable of executing 9 instructions in parallel. The functional units include:

- 12x12 multiply-accumulate
- 1 ALU
- 1 Shift Unit

- 32x12 Static register file
- 32x12 Local memory with support for 2 modulo buffers

Processors in the array exchange data with nearest neighbors through blocking FIFO channels, meaning the progression of algorithms in each processor is regulated by the availability of incoming data, or else vacancies in the channel for outgoing data. Such a dataflow communication and synchronization scheme is a special case of the well-known Kahn process network [8]. Unterminated FIFO channels at the accelerator border can be directly connected to the high data rate streams of data converters, or other dataflow-synchronized device. The efficient and low overhead communication scheme allows the processor to handle data-rates on the order of the processor clock speed. An AMBA interface, with a direct-cell-access network connection to each processor, is provided to enable configuration download, as well as low rate data exchange with the hardware system.

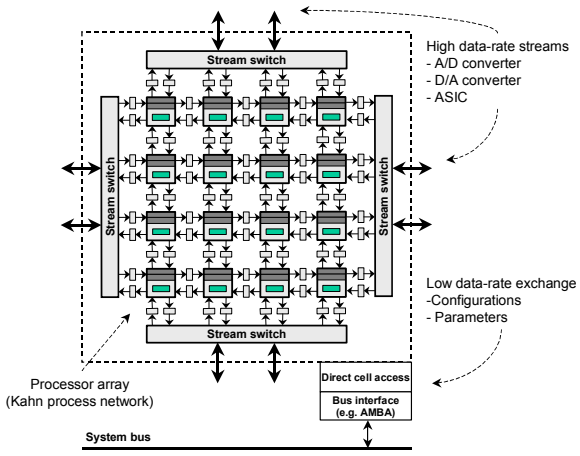

Figure 6: Architecture of a 4x4 cell configuration of the Bresca accelerator.

The number of processors in a Bresca accelerator is paramaterizeable. Including interconnection and control overheads, one cell occupies an average of 0.4mm^2 in 0.13µ CMOS technology. A fully utilized Bresca cell will dissipate up to 0.17mW/MHz.

2.6. Bresca extensions to the Programming Flow

Since the Bresca accelerator has been generated from the same processor generation flow as AVISPA, the same compiler and simulation methodology is fully applicable to BRESCA; however, the multi-processor configuration introduces the issue of partitioning applications across multiple processors. In the case of stream based processing in a demodulator, DSP kernels are assembled hierarchically from a library of single cell programs. One example of a single cell program is that implementing 4 taps of a complex FIR filter. This cell program can be multiply cascaded to build larger filters.

The SHAPE array programming, compilation, and assembly environment facilitates design and evaluation of hierarchical array designs. In this environment multi-cell programs can be composed, compiled, and debugged, taking advantage of the tool's ability to read and interpret the original processor description for the accelerator and its cells. Figure 7 illustrates the view of a hierarchical DSP kernel design viewed using SHAPE.

On the scale of the demodulator, hierarchical multi-cell functions can be placed and connected on the array. In an analogy to placement, floorplanning, and interconnection of pins for VLSI design objects, DSP kernels and their boundary FIFO channels (or pins) are placed and interconnected on the Bresca accelerator. The SHAPE compacting utility automatically reshapes and replaces components of the hierarchical design to optimize cell utilization, and control pin locations.

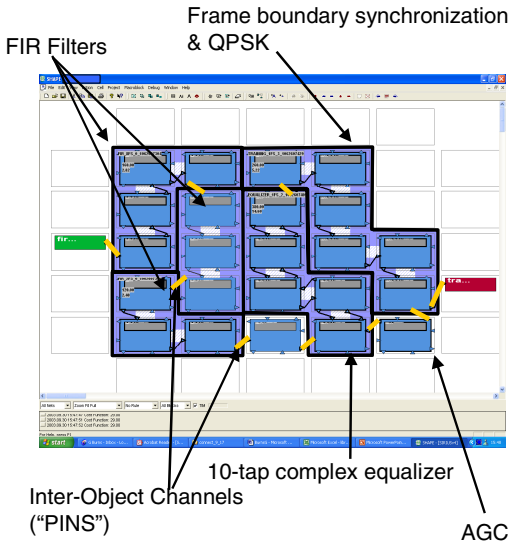

Figure 7: DSP kernel integration on BRESCA, using the SHAPE environment.

3. APPLICATION TO MULTI-STANDARD DIGITAL RADIO

In the domain of digital radio, the Bresca and Avispa accelerators enable software definition of the major digital radio receivers within a single system-on-chip. Real-time and functional requirements are approximately defined by the needs of SDARS, which specify the simultaneous demodulation of three parallel transmission streams,

including two QPSK streams and one OFDM stream [4][5]. In the receiver, all demodulated signals are combined and undergo error correction. Additional standards, such as DAB, are to run using a subset of these resources.

3.1. Hardware System

The multi-standard digital radio hardware configuration is illustrated in Figure 8 below. One Bresca and Avispa accelerator are integrated with one ARM processor, memory, and peripherals on an AMBA subsystem. Two on-chip ADCs are included, which can be directly connected to the input stream of the Bresca accelerator. The Bresca accelerator must contain sufficient cells to simultaneously demodulate two QPSK streams, as well as the time-domain processor of one OFDM stream. In such an application a 64-cell Bresca array is required.

An ASIC design methodology has been used to develop this system in 0.13µ CMOS. A target clock speed of at least 150MHz is guaranteed under worse case (military) process conditions. The potential of subsequent results to benefit from semi-custom or custom design VLSI design methodologies is significant, although not evaluated in this paper.

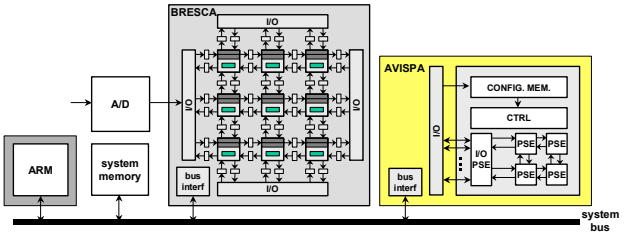

Figure 8: Hardware system for a multi-standard digital radio IF/baseband IC, capable of receiving either of the satellite digital audio radio services, as well as the major terrestrial digital broadcast formats.

3.2. Library

DSP tasks for QPSK demodulation, OFDM demodulation, and error correction were compiled onto the appropriate accelerator. Table 1 shows the attributes of representative DSP functions in the library, which were compiled onto the Bresca stream accelerator. The DSP Inner Loop cycle count represents the quantity of cycles the array-program requires per input sample. The cells column indicates the number of Bresca cells over which the algorithm is partitioned, and the energy column represents the energy dissipated per complex input sample. To calculate the mW of power dissipated by each library element on Bresca, one would multiply the incoming sample rate in MHz by the energy dissipated in nJ.

One of the important tradeoffs made in designing an algorithm, and which is enabled by this architecture, is between the size of the DSP Inner Loop versus the number of cells over which the algorithm is partitioned. Typically the number of inner loop cycles is minimized if the incoming sample rate approaches the clock speed of the processor. For the case of low sample rate relative to the processor clock speed, the designer will prefer to partition more functionality per processor cell to minimize the amount of resources consumed. In this regime processor partitioning is limited by the storage resources on the cell.

DSP Task	DSP Inner Loop	Cells	Energy
FIR 26-tap semi-cplx, 4x decimate	4	7	5
SQRC, 39-tap sc, 2x decimate	16	3	8
LMS equalizer, 10-tap complex	21	5	9
In-band AGC	22	1	2
Frame-boundary correlator (24-symbol matched filter)	24	4	8
Peak detector	24	1	2
Lock-state & QPSK decider	24	1	2
	(cycles)		(nJ/sample)

Table 1: Attributes of select Bresca streaming library elements

Table 2 shows the attributes of representative DSP functions that were compiled onto the Avispa block accelerator. The cycles column represent the processor cycles required to compute one OFDM symbol of such operations. In the case of the non-FFT functions, this result pertains to processing the algorithm on 1K complex samples framed in memory. The energy in mJ provides the dissipation to process the algorithm on one OFDM symbol. Power consumption in mW can then be calculated by multiplying the value in µJ by the OFDM symbol rate in kHz.

DSP Task	Cycles	Energy µJ
(I)FFT 2K Complex	11K	9.6
1K Inter-Symbol Differential Demod	1.1K	0.9
1K QPSK Demapper	1.1K	0.9
Viterbi Decoder K=9, r=1/2	17K	14.5
	per OFDM Symbol	µJ/Symbol

Table 2: Attributes of select Avispa library, relevant to one 1K OFDM symbol of data

3.3. Benchmarking of results

The silicon area of the multi-standard system-on-chip, excluding the pad ring, is indicated in Figure 9 below. In this system, the audio decoder is implemented on the ARM. Approximately 30mm2 is attributed to the programmable multi-standard receiver. Power consumption of the multi-standard receiver, alone, is less than 900mW, when both QPSK and single OFDM demodulators are simultaneously active.

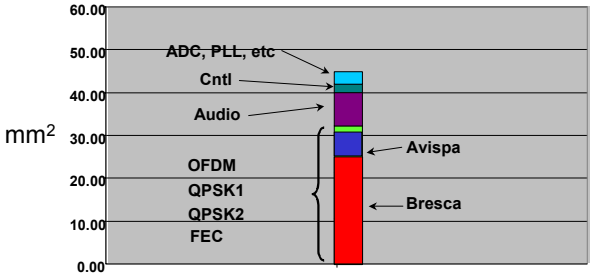

Figure 9: Summary of silicon area occupied by the Multi-Standard platform IC, along with expected area occupation of each platform element when fabricated in 0.13µ CMOS ASIC technology. The rightward bar indicates area occupied by Avispa and the 64-cell Bresca.

10. REFERENCES

[1] J. Leijten, A *Massively Parallel Reconfigurable ULIW Core*, Microprocessor Forum, San. Jose, October 12, 2003.

[2] J. Leijten, G. Burns, J. Huisken, E. Waterlander, and A van Wel, AVISPA: A Massively Parallel Reconfigurable Accelerator, System-on-Chip 2003 Conference, Tampere, Finland, November, 2003.

[3] P. Gruijters & G. Burns, *System-on-Chip Design Tutorial*, SDR Technical Forum, Orlando, November, 2003.

[4] F. Davarian, Sirius Satellite Radio, Radio Entertainment in the Sky, IEEE Aerospace Conference Proceedings, Vol 3., pp 1031-1035, 2002.

[5] Stellos Patsiokas, XM Satellite Radio Technology Fundamentals, SAE 2001 World Congress, Detroit, MI, March, 2001.

[6] D. Layer, *Digital Radio Takes to the Road*, IEEE Spectrum, July, 2001.

[7] See www.eurekadab.org.

[8] Lee & Parks, *Dataflow Process Networks*, Proc. IEEE 83, No. 5, pp 773-801, 1995.