HETEROGENEOUS RECONFIGURABLE SYSTEMS

Jan M. Rabaey
Arthur Abnous, Yuji Ichikawa†, Katsunori Seno‡, Marlene Wan

EECS Dept., University of California, Berkeley, CA
† Sharp Corporation, Tenri, Japan
‡ Sony Corporation, Tokyo, Japan

Abstract — The continually increasing integration density of integrated circuits portrays important paradigm shifts in next-generation designs, especially in the direction of systems-on-a-chip. Hybrid architectures mixing a variety of computational models are bound to be integrated on a single die. This opens the door for creative high-performance low-energy solutions to the programming problem using techniques such as reconfiguration to construct optimized architectures for a given computational problem. Exploiting the opportunities offered by these architectural innovations obviously requires a clear understanding of the trade-off's offered by the various architectural models and styles, as well as a well-thought out design methodology, combining high-level prediction and analysis tools with partitioning, optimization and mapping techniques. This paper presents an overview of opportunities of these reconfigurable architectures in the architecture domain.

INTRODUCTION

Systems-on-a-chip are becoming a reality at this very moment, combining a wide range of complex functions on a single die. Integrated circuits that merge core processors, DSPs, embedded memory, and custom modules have been reported by a number of companies [Borel97]. Projections of future integration densities suggest that this trend will surely continue in the next decade. This is illustrated in Fig. 1., which plots projections into the year 2012 of expected chip sizes as well as transistor densities for microprocessors, ASICs, and DRAMs [SIA97].

It is therefore by no means a wild conjecture to assume that a future generation design will combine all the functionality of a mobile multimedia terminal, including the traditional computational functions and operating system, the extensions for full multimedia support including graphics, video and high quality audio, and wired and wireless communication support. In short, such a design will mix a wide variety of architecture and circuit styles, ranging from RF and analog to high-performance and custom digital (Fig. 2.).

Such an integration complexity may seem daunting to a designer and might make all our nightmares regarding performance, timing and power come true. On the other hand, the high level of integration combined with its myriad of design choices might be a blessing as well and can effectively help us to address some of the most compelling performance, energy or power-dissipation problems facing us today.

0-7803-3806-5/97/$10.00 ©1997 IEEE

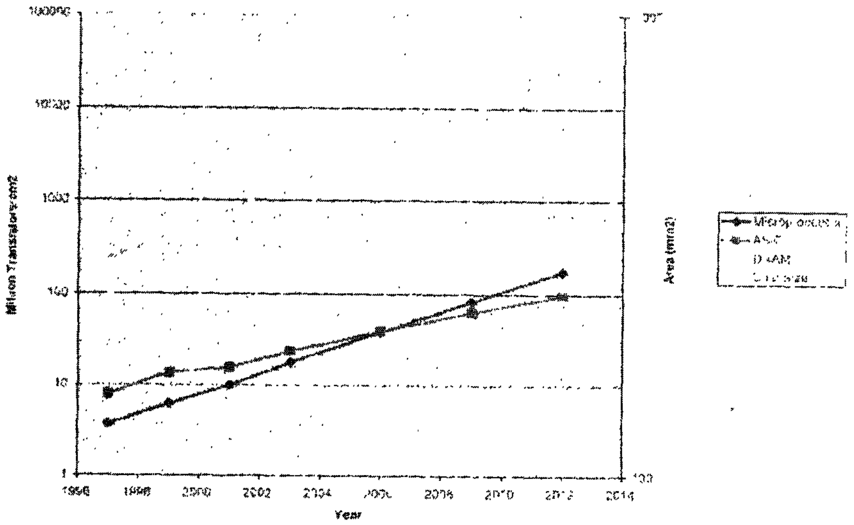


Fig. 1. Projected evolution in transistor density for microprocessors, ASICs, and DRAMs as well as chip sizes [Source: SIA97].

Exploiting these opportunities requires a system-oriented design methodology, which is certainly not in place today.

When considering the options these systems-on-a-chip offer, one of the first question that comes to mind is how these chips will be architected. The design and fabrication costs of these billion-transistor parts has the potential to be astronomical. This seems to suggest a trend towards fewer, more programmable designs that can span a wide range of applications. The ultimate would be the ultra-high performance microprocessor-system-on-a-chip that has the compute power to handle most applications, and is surrounded with enough programmable logic to provide the necessary I/O functionality. While attractive, this model has some important pitfalls. It ignores the fast that most embedded applications — which is where the majority of these circuits will be used — have some stringent requirements not only

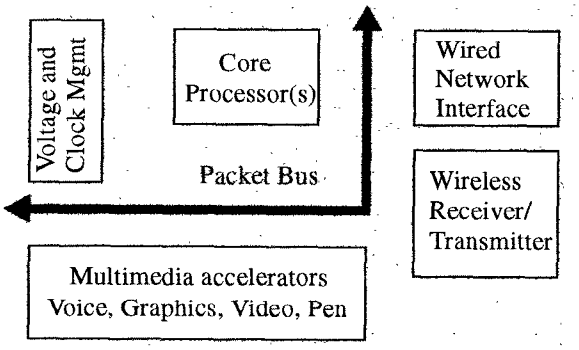

Voltage and Clock Mgmt
Core Processor(s)
Wired Network Interface
Packet Bus
Wireless Receiver/ Transmitter
Multimedia accelerators Voice, Graphics, Video, Pen

Fig. 2. Multimedia terminal on-a-chip

in performance but also in cost and power. Secondly, the functions that compose an integrated system such as the multimedia terminal are sufficiently different that they warrant being mapped onto different implementation fabrics. This is for instance demonstrated in Fig. 3., which plots the projected performance trends of general purpose processors versus multimedia processors [Sasaki96]. While general purpose performance is seen to level off in the coming years, the speed of multimedia processors will continue to increase exponentially. The difference between the two is that multimedia possesses an inherent parallelism that is not present in general purpose computing, where the average concurrency has been determined to be no higher than 4. Similar arguments can be used to demonstrate the special needs of other application areas such as communications, graphics, or automotive.

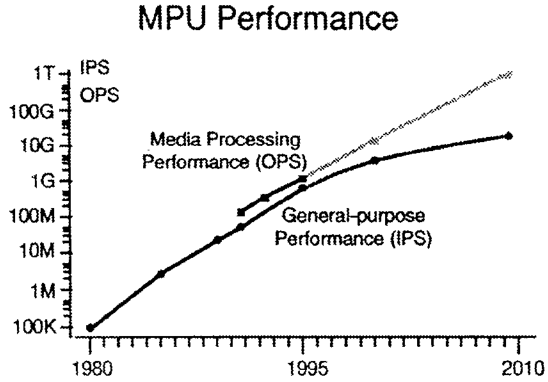


Fig. 3. Performance of general-purpose processors and multi-media processors, as projected over time [Sasaki96].

In this paper we will analyze the need for integrating a variety of programmable architectures on the same die, and discuss the impact of doing so on the performance-energy-area cost functions and the design methodology. While the paper will mostly focus on power (energy) dissipation as the main cost function to be optimized, similar arguments can be made for other cost functions such as performance.

PROGRAMMABLE ARCHITECTURES — AN OVERVIEW

It is generally agreed that dedicated custom implementations yield the best solutions in terms of the traditional cost functions such as performance, power, and area (PDA). Indeed, it is hard to beat a solution that is optimized to perform solely a single well-defined task. However, it is also realized that time-to-market and flexibility are also important assets. Many applications tend to require multi-functionality (for example, the multi-modal radio), or adaptivity. Very often, the specification of application tends to evolve during the design time and even after the part has been shipped for fabrication. This forms the lure for the so-called programmable solutions, which trade-off PDA for flexibility and (re)programmability. For a long time, programmable architectures have been narrowly defined to be of the *load-store* style processors, either in stand-alone format, or in clusters of parallel operating units (SIMD, MIMD). The latter have traditionally been of the homogeneous type, i.e. all processing units are of the same type and operate on the same type of data. In recent years, it has been observed at numerous sites that this model is too confining and that other programmable or *configurable* architectures should be considered as well. This was inspired by the success of programmable logic (FPGA) to implement a number of computationally intensive tasks at performance levels or costs that were substantially better than what could be achieved with traditional processors [Villa97]. While intrinsically not very efficient, FPGAs have the advantage that a computational problem can be directly mapped to the underlying gate structure, hence avoiding the inherent overhead of fixed-word length, fixed-instruction-set processors. Configurable logic represents an alternative architecture model, where programming is performed at a lower level of granularity.

Architecture Models

Trading off between those architectures requires an in-depth understanding of the basic parameters and constraints of the architecture, their relationship to the application space, and the PDA (power-delay-area) cost functions. While most studies with this respect have been either qualitative or empirical, a quantitative approach in the style advocated by Hennesy and Patterson for traditional processor architectures is desirable. Only limited results in that respect have been reported. The most in depth analysis on the efficiency and application space of FPGAs for computational tasks was reported by Andre Dehon [Dehon96], who derived an analytical model for area and performance as a function of architecture parameters (such as data-path width *w*, number of instructions stored per processing element *c*, number of data words stored per processing element *d*), and application parameters

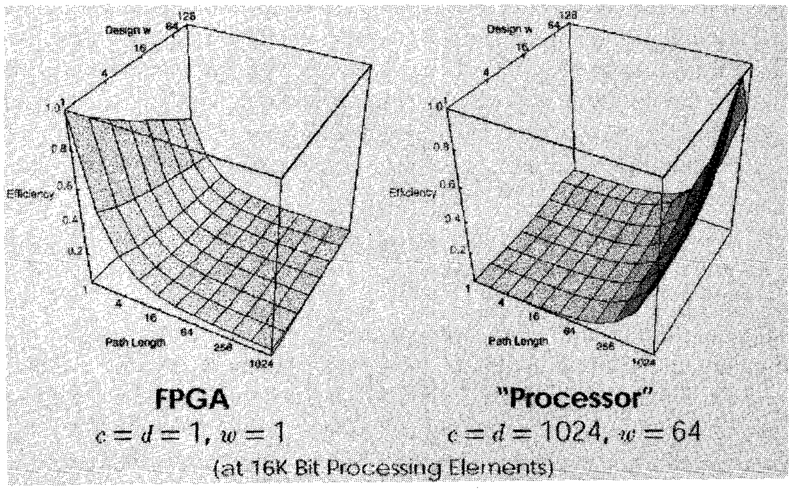


Fig. 4. Efficiency of FPGAs versus processors. Parameters are the word length *w*, the path length *d* and the contexts *c* [from Dehon96].

(such as word length w_a and path length l_p — the number of sequential instructions required per task). Fig. 4. plots one of the resulting measures of the model, the *efficiency* — the ratio of the area of running an application with word length w_a on an architecture with word length w_a versus running it on architecture with word length w. As can be observed, the processor excels at larger word lengths and path lengths, while the FPGA is better suited for tasks with smaller word and path lengths.

Limiting the configurable architecture space to just those two architectures has proven to be too restrictive and misses major opportunities to produce dramatic reductions in the PDA space. Potential expansions can go in a number of directions:

- By changing the architecture word length w — sharing the programming overhead over a number of bits. This increases the PDA efficiency if the application word length matches the architecture word length.
- By changing the data storage d — this introduces the potential for local buffering and storing data.
- By changing the number of resources r — this makes it possible to implement multiple operations on the PE by providing concurrent units (programming in the space domain).
- By changing the number of contexts c — this makes it possible to implement multiple operations on the PE by time-multiplexing (programming in the time domain).
- By reducing the *flexibility f*, i.e. the type of operations that can be performed on the processing element.

Definitions:

- The *flexibility index* of a processing element (PE) is defined as the ratio of the number of logical operations that can be performed on the PE versus the total set of possible logical operations. PEs that can perform all logical operations, such as general-purpose processors and FPGAs, have a flexibility index equal to 1. Dedicated units such as adders or multipliers have a flexibility close to 0, but tend to score considerably better in the PDA space.
- The *granularity index* of a processor is defined as a function $g(w,d,r,c)$, which is a linear combination of w, d, r, and c parameters, weighted proportionally to their cost.

A number of authors have considered various combinations of the above parameters. For instance, Dehon [Dehon96] advocated the introduction of multiple contexts in an FPGA architecture. The PADDI architectures, introduced at UC Berkeley [Chen92, Yeung95] and targeting the rapid prototyping of high performance applications, increased w, d, and c. For instance, these parameters were set to 16, 6, and 8, respectively, in the PADDI-2 architecture. An interesting study is presented at the 1997 Sips workshop in [Lieverse97]. The authors compare the efficiency of configurable architectures as a function of the granularity of algorithm (in terms of the complexity of the basic operators). The study was performed over a set of 21 video algorithms. They report that for this set of benchmarks, a larger area efficiency is obtained when using more complex operators on a smaller number of large granularity PEs.

Most of these studies ignore the impact of changing the flexibility index, which can have an enormous impact on the PDA cost function. This is illustrated by the example of a correlator for a CDMA radio, whose block diagram is shown in Fig. 5. When comparing the energy needed to perform the operation on a variety of PEs, dramatic differences can be observed, as shown in Table 1. The FPGA solution is an order of magnitude more efficient in energy than the general purpose processor (both architectures have a flexibility index of 1). This difference can mostly be attributed to the mismatch between granularity of application and architecture. On the other hand, optimizing the processing element for a single function (the ASIC approach) results in another improvement by more than two orders of magnitude! This dramatic reduction in energy argues that reducing the flexibility is an option that should not be ignored when delineating the architectural space for the future systems-on-a-chip.

Table 1. Impact of architectural choice on energy dissipation for CDMA correlator.

CDMA correlator	ARM 6 (5V, 20 MHz)	Xvlinx 4003 (5V, 64 MHz)	ASIC (1.5V, 64 MHz)
Energy	2765 nJ	394 nJ	1.2 nJ
Energy-Delay	167736 fJsec	394 fJsec	1.2 fJsec
Energy-Delay (normalized to 5V)	167736 fJsec	394 fJsec	1.04 fJsec

This brings us to the next level of architectural modeling, the composition.

Homogeneous and Heterogeneous Architectures

An overall chip architecture can be considered as a composition of computing elements with varying degrees of granularity and flexibility. This brings another set of parameters into the model: homogeneity and connectivity.

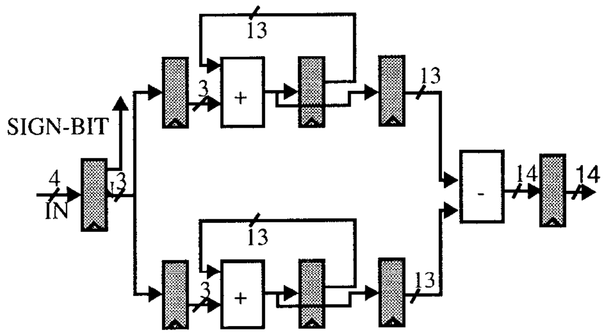


Fig. 5. Block diagram of correlator.

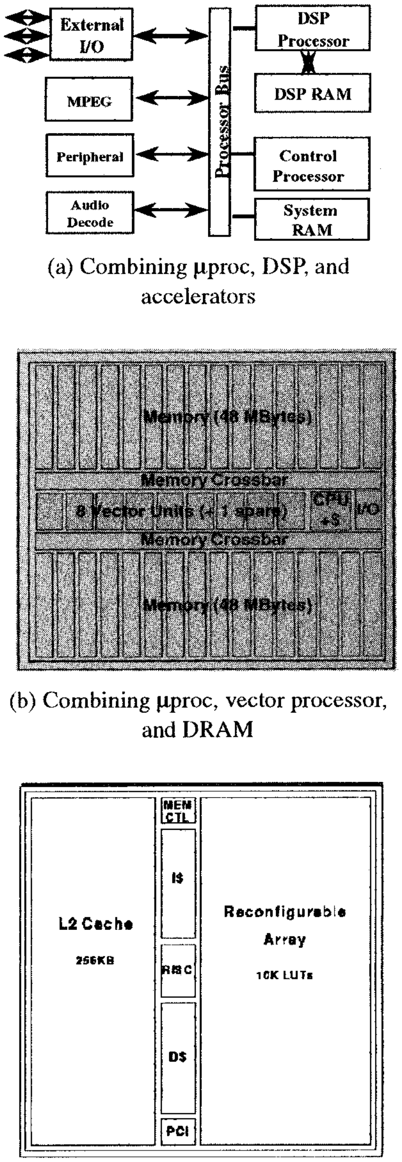
- An architecture is called *homogeneous* if the composing PEs are identical. This has been the architecture of choice for the majority of the multi-PE architectures so far. Examples are multi-processors (such as PADDI-2) and FPGAs. Maintaining homogeneity tends to improve processor utilization and simplifies the mapping problem. On the other hand, embedded systems seem to embrace heterogeneity. This is mainly due to the diversity in the computational requirements of a typical system. The multimedia terminal of Fig. 2., for instance, combines a wide variety of functions, each with different degrees of granularity, adaptivity, and type of operations.
- The *connectivity* determines the degree of interconnection between the PEs, and impacts the PDA and the flexibility of the overall architecture. Word length, context, resource, and flexibility parameters can be defined for the interconnect as well. An in-depth discussion of interconnect and its parameters is out of the scope of this paper.

Based on the above analysis, it is possible to classify future systems-on-a-chip into three categories:

- Homogeneous arrays of general-purpose processing elements. Architectures are differentiated by the granularity of the processing elements. The only departure from the overall homogeneity is that these parts typically will include large chunks of embedded memory. It is for instance projected that FPGAs in the year 2010 can pack between 2 and 5 million "real" gates and will contain more than 1 Mbyte of memory. Circuits of this class are typically used for general purpose computations and prototyping with limited constraints in the PDA domain.
- Application-specific combination of processing elements. Implementations of these types are typically geared towards a single application. They act as *board replacements*, and combine flexible components with application-specific accelerators. The implementation of these dedicated systems only makes economical sense for large volumes.
- Heterogeneous combinations of processing elements of different granularity and flexibility. These represent the real novelty in the system-on-a-chip era, and can be called under the denominator of *agile computing systems* [ISAT97]. The heterogeneity by nature restricts the applicability of the circuit to a limited domain (*domain-specific processors*), but at the same time yields solutions that score well in the PDA space. The most important question to be answered by the would-be designer is the choice of the programming elements and their connectivity.

The remainder of this paper will be devoted to the latter category. The possible trade-off's will be discussed based on a comparison between a number of emerging approaches. One architectural template, proposed in the Berkeley Pleiades project, will be discussed in more detail.

Agile Computing Systems (heterogeneous compute systems-on-a-chip)



(a) Combining μproc, DSP, and accelerators

(b) Combining μproc, vector processor, and DRAM

(c) Combining μproc and FPGA.

Fig. 6. Heterogeneous, programmable systems-on-a-chip.

As mentioned earlier, a range of system-on-chip implementations have already been realized by industry. Most of these are of the type, depicted in Fig. 6.a, i.e. a combination of a microprocessor subsystem (e.g. ARM 8), DSP processor (TI2x), and dedicated accelerator units (MPEG or audio decoders), connected through a standard processor bus. While having the advantage of being composed of well-understood building blocks with established software support, the overall combination does not yield dramatic improvements in the PDA space, is restricted in its application domain, and is hard to program as no overlaying computational model is defined.

An interesting combination of a general purpose processor, a vector processor and large blocks of DRAM was proposed in the Berkeley IRAM project [Patter97]. This heterogeneous combination of processor architectures (Fig. 6.b) has the advantage that the overall model is well understood from the super-computing era, and that system software is readily available. Applications can easily be identified in the graphics and multimedia areas. An IRAM processor implemented in a 0.18 μm technology and clocked at 1 GHz is projected to provide 16 GFLOPS (64 bit), 128 GOPS (8 bit), and 96 Mbit of DRAM.

The combination of microprocessor and FPGA (Fig. 6.c) has achieved a lot of attention recently [Brass97, Napal000]. The FPGA can serve for a variety of functions, such as extending the instruction set of the core processor, implementing a high-performance dedicated compute engine, or as peripheral unit. Software support is once again the main hurdle for this system approach to overcome. To be successful, fast, predictable, and verifiable compilation is a necessity. It is furthermore not clear at the time of writing how much impact this approach has in terms of addressing the PDA constraints of embedded systems.

THE BERKELEY PLEIADES PROJECT

The heterogeneous architectures presented above cover two, or at most three spots in the granularity/flexibility space. For instance, the structure of Fig. 6.c allows for a trade-off between either very small or very large granularity (each of which is completely flexible). It was stated earlier that typical system applications tend to present a variety of granularity requirements and that reducing the flexibility for certain functions can lead to major PDA improvements. The Pleiades architecture, under development at UC Berkeley [Abnous96, Rabaey97], attempts to integrate a wider variety of reconfigurable components into a single structure. The architecture, presented in Fig. 7. presents a reusable template that can be used to implement a domain-specific processor instance, that can then be programmed to implement a variety of algorithms within a given domain of interest. All instances of the architecture template share a common set of control and communication primitives. The type and the number of processing elements may vary; they depend upon the properties and the computational requirements of the particular domain of interest.

The architecture is centered around a reconfigurable communication network. Communication and computation activities are coordinated via a distributed data-driven control mechanism. Connected to the network are an array of heterogeneous, autonomous processing elements, called *satellite processors*. These could fall into any of the reconfigurable classes: a general microprocessor core (most of the time only one of these is sufficient), a dedicated functional module such as a multiply-accumulator or a DCT unit, an embedded memory, a reconfigurable data path, or an embedded PGA. Observe that each of the satellite processors has its own autonomous controller, although the instruction set of most of these modules is very shallow (i.e. weakly programmable).

The microprocessor core plays a special role. Besides performing a number of mis-

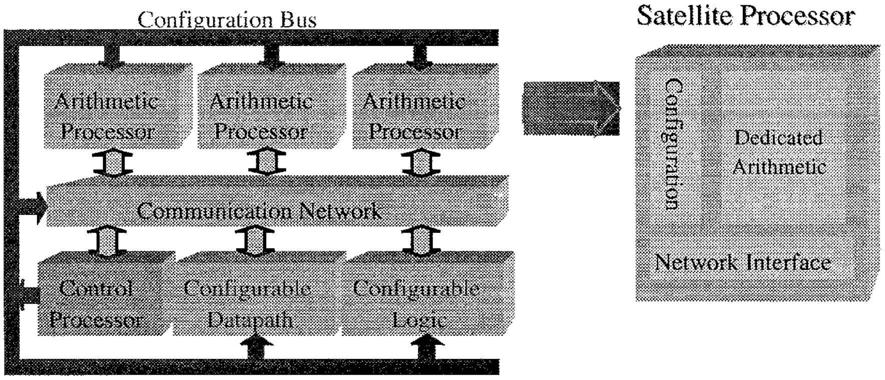


Fig. 7. Heterogeneous reconfigurable architecture, as defined in Pleiades.

cellaneous non-compute intensive or control-dominated tasks, it configures the satellite processors and the communication network (over the reconfiguration bus). It also manages the overall control flow of the application, either in a static compiled order, or through a dynamic real-time kernel.

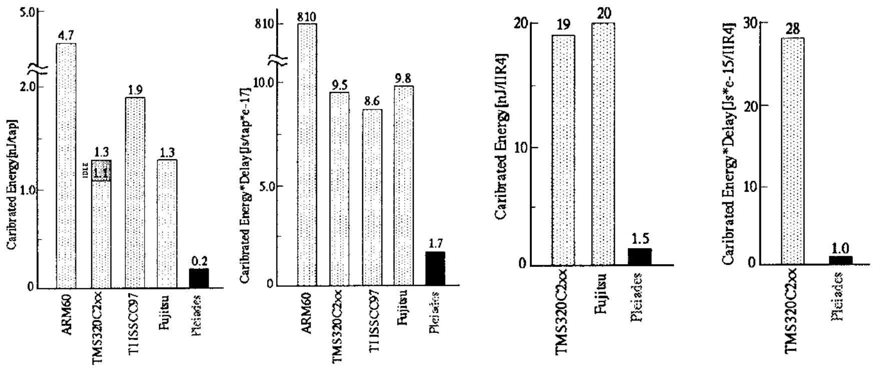
The application is partitioned over the various computational resources, based on granularity and recurrence of the computational sub-problem. For instance, a convolution is mapped onto a combination of address generator, memory, and multiply-accumulate processors. The connection between these modules is set up by the control processor and remains static during the course of the computation. The same modules can in another phase of application be used in a different configuration to compute, for instance, an FFT.

The Pleiades approach has the advantage that it can exploit the right levels of granularity and flexibility as needed by the application domain, yet that it can be supported by a well-defined design and implementation methodology.

Two simple benchmarks (FIR and IIR) help to illustrate the impact of these choices on the PDA cost functions, in casu energy and energy-delay. The satellite processors selected for the Pleiades instance used in this analysis include address generators (reconfigurable data paths), SRAM memories, and multiply-accumulators. Savings of more than two order of magnitude can be observed with respect to a traditional processor (ARM). Even when comparing to DSP processors, which are optimized for these tasks, savings between 5 and 30 in energy-delay product — which compares the efficiency of an implementation in the energy-performance domain — can be observed.

SUMMARY

Agile computing architectures will play a dominant role in the system-on-a-chip era. They combine the advantages of programmability, hence leveraging off the



(a) FIR Filter

(b) IIR Filter

Fig. 8. Energy and Energy-Delay numbers for FIR and IIR filters, implemented on a variety of architectures. Numbers are normalized with respect to voltage and technology.

design cost of a complex part over a number of designs and providing at the same time adaptivity an flexibility, with the PDA efficiency of more dedicated architectures. Research into the composition of these heterogeneous reconfigurable structures has just started and has some long way to go. Most importantly, quantifiable models of the impact of architectural parameters such as flexibility and granularity on the PDA costs and their relationship to the application parameters have to be developed. These will translate into a number of estimation tools that form the underpinning of a co-design methodology for heterogeneous architectures. The lack of an established implementation methodology is by far the most important impediment to the success of agile architectures.

ACKNOWLEDGEMENTS

This research is sponsored by DARPA as a part of its Adaptive Computation Systems initiative. Also appreciated are the sponsorship of Rockwell, Cadence, Motorola, Sharp and Sony.

REFERENCES

[Abnous96] A. Abnous and J. Rabaey, “Ultra-Low-Power Domain-Specific Multimedia Processors,” Proceedings 1996 VLSI Signal Processing Workshop, pp. 459-468, San Francisco, October 1996.

[Borel97] J. Borel, “Technologies for Multimedia Systems on a Chip”, *Proc. IEEE ISSCC Conference 1997*, pp. 18-21, San Francisco, February 1997.

[Brass97] *Berkeley Reconfigurable Architectures, Systems and Software (BRASS)*, http://www.cs.berkeley.edu/Research/Projects/brass.

[Chen92] D. Chen and J. Rabaey, “A Reconfigurable Multiprocessor IC for Rapid Prototyping of Real Time Data Paths”, *Proc. IEEE ISSCC Conference 1992*, pp. 56-57, San Francisco, February 1992.

[Borel97] J. Borel, “Technologies for Multimedia Systems on a Chip”, *Proc. IEEE ISSCC Conference 1997*, pp. 18-21, San Francisco, February 1997

[Dehon96] A. DeHon, “Reconfigurable Architectures for General Purpose Computing,”, Technical Report 1586, MIT Artificial Intelligence Laboratory, September 1996.

[ISAT97] *Silicon after 2010*, DARPA ISAT study group, August 1997.

[Lieverse97] P. Lieverse et al., “A Clustering Approach to Explore Grain Sizes in the Definition of Weakly Programmable Processing Elements”, Proceedings 1997 Sips Workshop, Leicester, November 1997.

[Napa1000] *National Semiconductor's Adaptive Systems on-a-Chip*, http://www.national.com/appinfo/milaero/napa1000.

[Patter97] D. Patterson et al., “Intelligent RAM (IRAM): Chips that Remember and Compute”, *Proc. IEEE ISSCC Conference 1997*, pp. 224-225, San Francisco, February 1997.

[Rabaey97] J. Rabaey, “Reconfigurable Computing — The Road to Low Power DSP“, Proceedings IEEE ICASSP Conference, Munich, April 1997.

[Sasaki96] H. Sasaki, “Multimedia Complex on a Chip,” *Proc. IEEE ISSCC Conference 1996*, pp. 16-19, San Francisco, February 1996.

[SIA97] *The 1997 SIA Roadmap*, http://www.sematech.roadmap.org/public/roadmap/index.html

[Villa97] J. Villasenor and W. Mangione-Smith, “Configurable Computing“, Scientific American, pp. 66-73, June 1997.

[Yeung95] A. Yeung and J. Rabaey, “A 2.4 GOPS Data-Driven Reconfigurable Multiprocessor IC for DSP”, *Proc. IEEE ISSCC Conference 1995*, pp. 108-109, San Francisco, 1995.