Trends in Reconfigurable Computing

(invited paper) Reiner Hartenstein, Dept. of Computer Science (Informatik),
University of Kaiserslautern, D-67653 Kaiserslautern, Germany

Abstract

The paper gives a survey on Reconfigurable Logic and Reconfigurable Computing as well as on recent R&D trends in these areas. The paper also points out the weak aspects of recent developments and proposes new directions to go.

1. Introduction

Rapidly increasing number and attendance of conferences on re-configurable computing (the three most important ones, FCCM, FPGA, and FPL ([1], the eldest and largest), attracted between 200 and 250 paying attendants in 2000 and later) and workshops (RAW, RSP, ENREGLE etc.) as well as the adoption of this topic area by congresses like ASP-DAC, DAC, DATE, ISCAS, SPIE, and many others indicate, that reconfigurable platforms are heading from niche to mainstream, supported by a rapidly growing large user base of HDL-savvy designers with FPGA experience. The echo on my double time slot embedded tutorials [2] [3] rapidly stimulated a number of further invitations. Reconfigurable platforms bring a new dimension to digital system development and have a strong impact on SoC Design. If the state of the art of the design flow would be as desired, their flexibility could support turn-around times of minutes instead of months for real time in-system debugging, profiling, verification, tuning, field-maintenance, and field upgrades.

2. FPGAs

FPGA Vendors stepping forward rapidly. FPGA vendors on the market are: Actel, Altera, Atmel, Cypress, Lattice, Lucent, Quicklogic, Triscend, and Xilinx (also see figure 1 b). The PLD market is poised to grow, according to many industry watchers. Currently FPGA vendors have a relatively fast growing large user base of HDL-savvy designers. Their ability to support also only limited products and platforms make design efforts to be more focused. Cost differences between volume FPGAs and (volume) ASICs are shrinking. Driven by a growing large user base innovations occur more and more rapidly. FPGA vendors are heading for the forefront of platform-based design. Altera and Xilinx are currently the leading FPGA vendors (figure 1), both with a volume of sales almost 1.5 Bio US-$ in the year 2000. Advantages of PLDs are becoming apparent to the marketplace. Dataquest calls programmable logic the fastest growing segment of the entire semiconductor market. Mostly driven by telecom and wireless-communications applications growing 20% annually, PLD revenue will jump to $7.04 billion in 2004 [IC Insights].

rank 2000	vendor	%
1	Xilinx	42
2	Altera	37
3	Lattice	15
4	Actel	6

Fig. 1: Top 4 FPGA manufacturers 2000 (total of 3.7 bio $)

Terminology is an important aspect since we are far from consensus. In Reconfigurable Systems (RS) we should be clearly distinguish (fig. 2) between the areas of Reconfigurable Logic (RL) also called field-programmable logic (FPL), and, Reconfigurable Computing (RC). RL, also called field-programmable logic (FPL), where the typical product name is FPGA (field-programmable gate array, sometimes called PLD: for programmable logic device), deals with fine-grained reconfigurable circuits and systems. A typical fine-grained reconfigurable circuit consists of an array of CLBs (configurable logic blocks) with a path width of 1 bit, which are embedded in a reconfigurable interconnect fabrics. From an EDA point of view this RL level appears as a methodology of logic design for hardwired logic, but "on a strange platform", which is not really hardwired. RC, where typical procucts are reconfigurable arrays (RA) or reconfigurable data path arrays (rDPA), deals with coarse-grained reconfigurable circuits and systems. A typical coarse-grained reconfigurable circuit consists of an array of CFBs (configurable functional blocks), also called reconfiburable datapath unit (rDPU), with a wide path width like, for instance, 16 or 32 bit. Here we may also say, the RA granularity is 16 or 32 bit. Also RAs with multiple granularity are known, where medium grain rDPU slices of pathwidth W (4 bits, for example) may be bundled for rDPUs with a path width of N x W, where N is a non-negative integer.

Meanwhile a wide variety of hardwired IP cores is delivered on board of the same chip with the FPG. Due to Moore's law the FPGA vendors offer more and more products having microcontrollers like ARM, MIPS, PowerPC, or other RISC architectures, memory, peripheral circuitry and others, together with the FPGA on board of the same chip (fig. 11 b). A Xilinx FPGA, for example, has 4 PowerPCs on board. and 24 Conexant 3.125 Gb/s serial link cores providing a total of 75 Gb/s/chip link bandwidth. Such a symbiosys between

term	granularity (path width)	configurable blocks
Reconfigurable Logic	fine grain (~1 bit)	CLBs: configurable logic blocks
Reconfigurable Computing	coarse grain (example: 16 or 32 bits)	rDPUs: reconfigurable data path units (for instance: ALU-like)
	multi-granular (supports slice bundling)	rDPU slices (example: 4 bits)

Fig. 2: How to avoid confusion with the term of “reconfigurable”

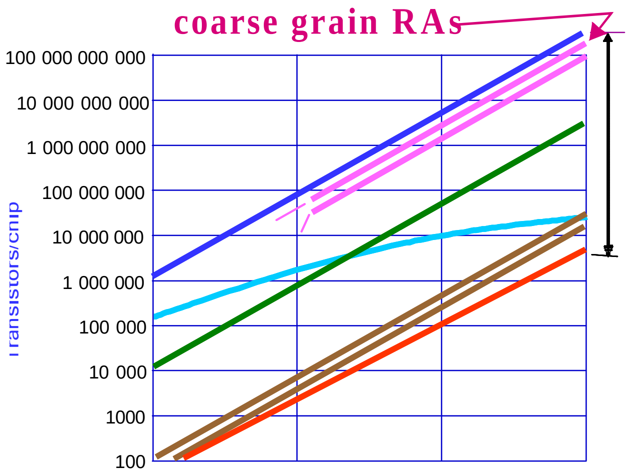


Fig. 3: Integration density of coarse grain reconfigurable computing arrays.

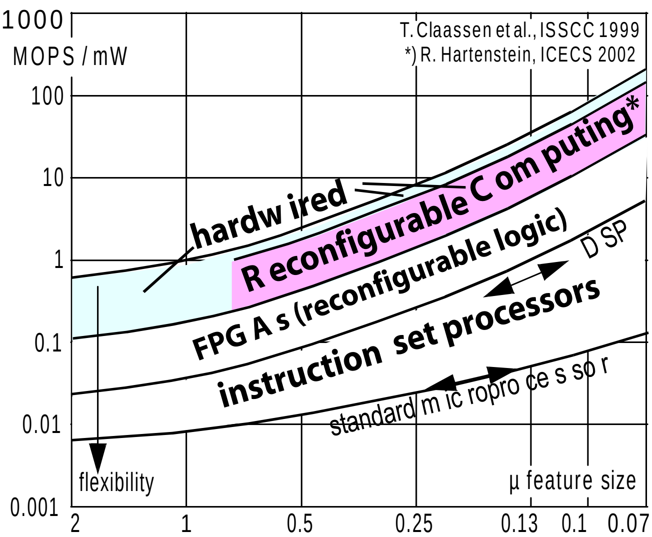


Fig. 4: Energy efficiency vs. flexibility including Reconfigurable computing.

FPGA and microprocessor correponds fig. 11 c. Including the interface hardware it corrsponds fig. 11 d.

2.1 Application and R&D areas

Mostly driven by telecommunication networks and wireless-communications applications a growth has been predicted of 20% annually, and FPGA vendor revenue will jump to $7.04 billion in 2004. In a number of application areas like multimedia, wireless telecommunication, data communication and others, the throughput requirements are growing faster than Moore's law [9]. The current state of the art in FPGAs does not yet provide sufficient performance. Other application areas are image processing digital signal processing, encryption, automotive electronics, etc. (also see [11] whtihin this conference proceedings volume). An emerging market is runtime reconfiguration, and, may be, or mav be not, evolvable hardware.

One of the eldest FPGA markets is for ASIC emulation. Rapid Prototyping / ASIC Emulation is an important application of FPGAs. Simulation is the most time consuming step during the IC design flow, which may take even days or weeks. This can be replaced by Rapid Prototyping: much faster emulation on large FPGA arrays. By acquisitions the 3 major EDA vendors offer ASIC emulators, along with compilers: has acquired Quickturn (Cadence), IKOS (Synopsys), and Celaro (Mentor Graphics), also offering such service over the internet. For smaller designs less complex emulation boards may be used, like Logic emulation PWB (based on Xilinx Virtex, can emulate up to 3 million gates), and, the DN3000k10 ASIC Emulator from the Dini Group. The area of FPGA use for system prototyping has its own international workshop series on Rapid System Prototyping (RPS) [5]

Run time reconfiguration (RTR) provides a powerful advantage of FPGAs over ASICs [4]: smaller, faster circuits, simplified hardware interfacing, fewer IOBs; smaller, cheaper packages, simplified software interfaces. Exploding design cost and shrinking product life cycles of ASICs create a demand on RA usage for product longevity. Performance is only one part of the story. The time has come to fully exploit their flexibility to support turn-around times of minutes instead of months for real time in-system debugging, profiling, verification, tuning, field-maintenance, and field-upgrades.

2.2 Evolvable hardware: a future market?

The terms "Evolvable Hardware" (EH), "Evolutionary Methods" (EM), sometimes also called "Darwinistic Methods", and biologically inspired electronic systems stand for a new research area, which also is a new application area of FPGAs [genetic FPGA]. It can be seen as a kind of revival of cybernetics or bionics, where the resurrection is stimulated by the new technology of reconfigurable hardware not having been available at past times. Currently most research goals are mainly based on using evolutionary methods (EM) and reconfigurable hardware platforms. The labelling „evolutionary“ and the „DNA“ metaphor helped to create a widely spread awareness and to raise research funds. Typical to the scene are freaks, who do almost everything with genetic algorithms, even when simulated annealing is by orders of magnitude more efficient. Also stimulated by research funding in the EU, in Japan, Korea, and the USA the EM-related scientific scenes and tracks, as well as many specialized international conferences are again mushrooming and difficult to survey. Shake-out phenomena should be expected, like those in the past with „Artificial Intelligence“ and other highly visionary scenes.

2.3 Efficiency

Due to higher degree of regularity the growth rate of FPGA integration density (number of transistors per chip) is higher than that of general purpose microprocessors (compare fig. 3). The growth rate is about the same as that of semiconductor memory. But this is only the physical integration density. But the logical integration density, i. e. of the parts which directly serve the application, is another factor of 100 behind, so that in total it is 4 orders of magnitude behind the Gordon Moore curve. (see fig. 3). Compared to memory and other full-custom-style integrated circuits, fine grain reconfigurable circuits are highly area-inefficient [6]. Due to rough estimations [7] only about one percent of the chip area serves the real application, whereas the other 99 percent are reconfigurability overhead. About 10% area are needed for storage of configuration code, and, about 90% area are needed for routing resources like wire pieces and switches. Nick Tredennick estimates, that for each transistor serving directly the application, about 200 more transistors are needed for reconfigurability.

ACRI
Alliant
American Supercomputer
Ametek
Applied Dynamics
Astronautics
BBN
CDC
Convex
Cray Computer
Cray Research
Culler-Harris
Culler Scientific
Cydrome
Dana/Ardent/Stellar/Stardent
DAPP
Denelcor
Elexsi
ETA Systems
Evans and Sutherland Computer
Floating Point Systems
Galaxy YH-1
Goodyear Aerospace
MPP
Gould NPL
Guiltech
ICL
Intel Scientific Computers
International Parallel Machines
Kendall Square Research
Key Computer Laboratories
MasPar
Meiko
Multiflow
Myrias
Numerix
Prisma
Tera
Thinking Machines
Saxpy
Scientific Computer Systems (SCS)
Soviet Supercomputers
Supertek
Supercomputer Systems
Suprenum
Vitesse Electronics

Fig. 5: Gordon Bell's "Dead Supercomputer Society" [keynote at ISCA 2000].

domain	environment	platform
time (procedural)	dataflow machines	indeterministic, dying R&D
	control-procedural programming (instruction stream-based computing)	von-Neuman-like machine architectures
time & space	data-procedural programming (data stream-based computing)	Xputer architectures using DPU arrays: coarse grain reconfigurable
space (structural)	logic design	hardwired
		configurable GAs
		FPGAs: re-configurable
		dynamically reconfigurable

Fig. 6: Segmentation of digital system implementation disciplines

The consequence of reconfigurability overhead is, compared to hardwired solutions, a higher power consumption (roughly by a factor of 10, see fig. 4) and lower switching speed or clock frequency (about a factor of 3 to 5). By re-design efforts reducing the clock speed a highly progressive improvement of power dissipation may be obtained, since reducing clock frequency by a factor of n yields a reduction of power dissipation by a factor of n^3 [8]. The design has to be re-optimized, since just tuning the clock would not yield this result. However, the reconfigurable solution is an order of magnitude more efficient than a software solution on a microprocessor (see fig. 4).

3. Coarse-Grained Reconfigurable Architectures

Since computational datapaths have regular structure potential, full custom designs of *reconfigurable datapath units* (rDPUs) are drastically more area-efficient (fig. 3). Coarse-grained architectures provide operator level CFBs, and very area-efficient datapath routing switches. A major benefit is massive reduction of configuration memory and configuration time, and drastic complexity reduction of the placement and routing problem. Several architectures are briefly outlined in [2].

Reconfigurable Computing stresses coarse grain reconfigurable arrays (RAs) with major pathwidths, like 16 or 32 bits, because fine-grained architectures are less efficient, due to a routing area overhead and routing congestion. Some arrays are multi-granular solutions supporting the bundling of resources, like e. g. from 4 ALUs of 4 bits each to obtain a 16 bit ALU. Since computational datapaths have regular structure potential, full custom designs of reconfigurable datapath units (rDPUs) can obtain a drastically higher integration density (see fig. 4), than by assembling it the FPGA way from single-bit CLBs. Also coarse grain reconfigurable arrays usually usually have microcontrollers (fig. 11 c) and hardwired interfacing circuitry on boad of the same chip (fig. 11 d). The model in fig. 11 e is not yet commercialized but is subject of R&D (section7).

Coarse grain reconfigurable circuits and systems show a igh area and energy efficiency. There is only a minor difference between the Gordon Moore curve and the physical integration density and even the logical integration density (figure 3). The energy efficiency practically reaches that of hardwired solutions (figure 4), which is about two orders o magnitude better than that with software on a microprocessor (figure 4). Coarse-grained architectures provide operator level CFBs (Configurable Functional Blocks), word level datapaths, as well as powerful and very area-efficient datapath routing switches, featuring massive reduction of configuration memory and configuration time, as well as drastic complexity reduction of the placement and routing problem, because only a few CFBs are needed. Using the same technology a coarse grain array implementation of the same algorithm is substantially faster than FPGAs.

3.1 Stream-based Reconfigurable Computing

Stream-based RC (with rDPU arrays: rDPAs) is urgently needed. In a number of application areas like multimedia, wireless telecommunication, data communication and others, the throughput requirements are growing faster than Moore's law [9]. The current state of the art in FPGAs does not yet provide sufficient performance. For flexibility and low power the only viable solution is one with rDPAs like offered by providers like PACT [10].

There are four different routes to implement DPAs. In design and implementation of embedded systems including distributed computing arrays (DPU arrays) several fundamentally different approaches are possible [11], coming along with different business models. The first approach uses fixed DPUs, which are not reconfigurable. All other solutions sketched below make use of reconfigurable DPU arrays synthesized from reconfigurable DPUs (rDPUs).

First, an application-specific solution is obtained from the usual design flow, but for (non-reconfigurable) DPAs (DPU arrays), where DPUs are designed directly or retrieved from a library, and, the flow is continued down to physical layout and tape-out. Fabrication is the last station of the flow, and we obtain a strictly application-specific hardwired product which is profitable only by very high production volume like for a mass market. The DPU array obtained by this approach is not reconfigurable.

Second, a fully universal coarse grain reconfigurable array using universal rDPUs might be the goal. But this approach does not seem to be very realistic, unless a massive wasting of resources is tolerated. The problem is illustrated already by the decision problem to select the

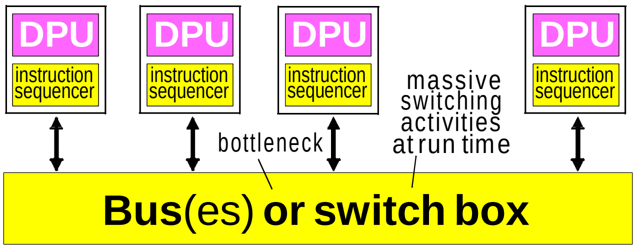


Fig. 7: Typical concurrent computing platform (memory not shown)

rDPU pathwidth: does a DPU need a pathwidth of 64 to become universal? Do all rDPUs need to have both, an integer multiplier and a floating point multiplier for universality. An attempt to come close to universality are arrays with multiple granularity which permits bundling several narrow path DPUs to a compound DPU featuring a wide datapath, like e.g. a 32 bit compound rDPU from 8 DPUs where each is 4 bits wide. Like the following approaches this solution creates a completely different business model, where personalization is carried out after fabrication. Due to high flexibility many different design can be implemented onto the same hardware platform.

Third, another solution is a domain-specific approach [2] [3], where rDPU architecture and other features are optimized for a particular application domain, like e. g. multimedia, wireless communication, or, image processing. Products from PACT Corp. are following this approach [10]. A design space explorer has been implemented to derive an optimum DPU array architecture from a benchmark or domain-typical set of applications within a few days [12] [13] [14]..

Fourth, a new solution is the soft array approach making use of soft rDPUs mapped onto a large FPGA (see section 6). This approach provides the highest flexibility. But its clock frequency may be a factor of about 2 to 3 slower than that of the three solutions mentioned above, which - if it is a problem at all - may be compensated by a higher degree of parallelism or a more clever design. For more details see section6.

EDA for DPU arrays or rDPU arrays is a key issue. All four routes have in common, that mainly the same design flow front end may be used for all of them. The tendency goes toward stream-based DPU arrays and there is no principle difference, whether the DPU array is hardwired or reconfigurable. The only important difference is binding time of placement and routing: before fabrication, or, after fabrication.

4. Stream-based instead of concurrent

The world of traditional informatics is based on computing in the time domain. Computing in time means, that a program is scheduling the microprocessor as a resource for instruction execution. Classical structures and principles in computing are von-Neumann-centric, partitioning the machine into datapath, instruction sequencer (instruction fetch and branching control), RAM, and, I/O (see fig. 11 a).

Let us look at the RAM-based success story. The "control-procedural" execution mechanism is modelled into the brain of each computing science student - as a common machine paradigm, with, or, without stack mechanism extensions. Due to the simplicity of this machine paradigm zillions of programmers can be educated. The hardwired processor can be fabricated in volume and all personalization (programming) is achieved by downloading the fully relocatable machine code into the immensely scalable RAM. The dominating instruction set became a quasi standard, where compatibility is managed from generation to generation of processors. Both, compatibility and being RAM-based, are the basis of the tremendous success of the software industry. This is the "normal" matter of computing.

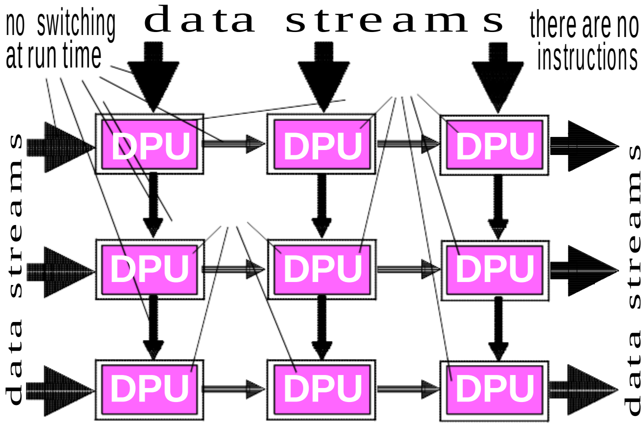


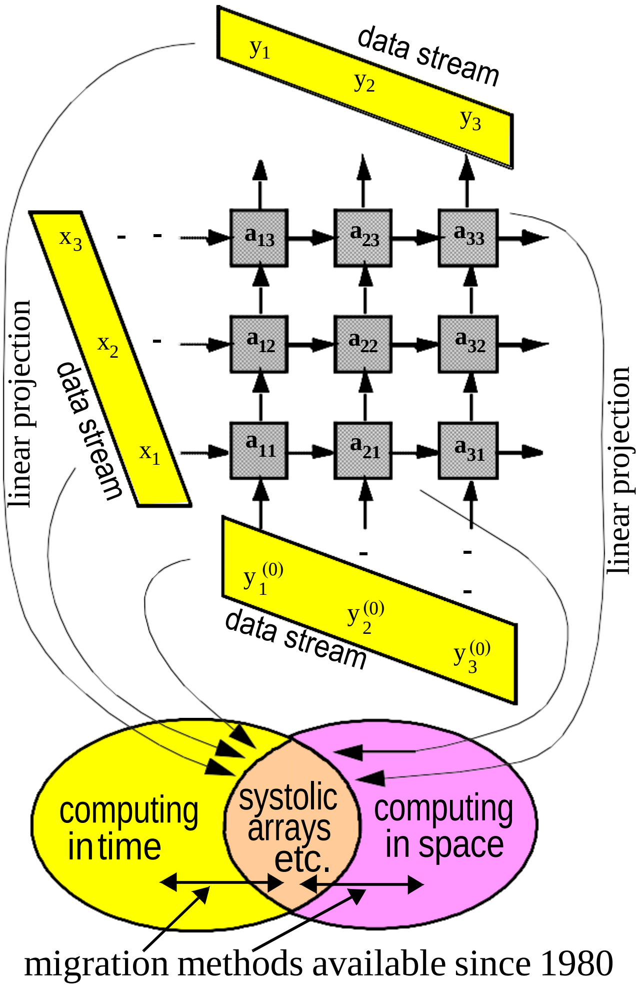
Fig. 8: Stream-based computing array example: transport-triggered exec.

But the world of computing in time (see fig. 6) has a lot of problems, like, for example, the processor / memory communication bottleneck, also called von Neumann bottle neck, which is widening from generation to generation and has reached about 2 orders of magnitude.

But also software processor solutions are inefficient relative to hardwired solutions (fig. 3 and fig. 4). There are fundamental flaws in software architectures. First, using time multiplexing with a single piece of logic. Secondly, the overhead associated in moving data back and forth between memory back and logic. Third, control itself is overhead. Fourth, pipelining in microprocessors adds another whole level of control overhead. Chips these days are almost all memory, and the reason is that the architecture is so wrong. Only about one percent of the power is going into real logic functions and 99 percent is going into caches and other hardware overhead [16]. It is shocking to find the difference as a factor of 100 to 1,000 [16], even to 10.000 [17]. The metric for what is a good solution has been wrong all the time. By hardwired solutions 1,000 MOPS per milliwatts or 1,000 MOPS per square millimeter can be obtained [16].

4.1 Parallel Computing vs. Reconfigurable

RISC core IP cells are available so small, 64 or more of them would fit onto a single chip to form a massively parallel computing system. But this is not a general remedy for the parallel computing crisis [18], indicated by rapidly shrinking supercomputing conferences and dying supercomputing industries (fig. 5). For many application areas process level parallelism yields only poor speed-up improvement per processor added. Amdahls law explains just one of several reasons of inefficient resource utilization. A dominating problem is the instruction-driven late binding of communication paths, which often leads to massive communication switching overhead at run-time (fig. 7). R&D in the past has largely ignored, that the so-called "von Neumann" paradigm is not a communication paradigm. However, some methods from parallel computing and parallelizing compiler R&D scenes may be adapted to be used for lower level parallelism on RA platforms. Processor architecture has reached a dead end with masses of research projects around execution pipelining and cache-related strategies, usually achieving only marginal improvements.



ig. 9: Systolic Arrays synthesis: programming in time and space.

The microprocessor is a Methusela. After almost a douzen of generations the microprocessor is a methusela. Not only in embedded systems the microprocessor more and more takes the role of a handicapped needing a wide variety of prostheses. Meanwhile the accelerators often occupy more silicon real estate than the processor core itself.

DPU arrays are more efficient than Concurrency. Classical parallel processing relying on concurrent processes is not the way to go, since its fundamental architecture relies on a uniprocessor, where pipelining brings only marginal improvements. Its alternative, (locally) distributed computing, i. e. DPU array computing, however, means parallelism by an application-specific pipe network in terms of multiple DPUs, not multiple CPUs (fig. 8). There's no CPU [19]. There's nothing "central". It's fully distributed, with lots of different DPUs containing adders, registers, multipliers -- just what's needed for direct mapping of the algorithm onto the architecture.

Reconfigurable Computing (RC) is the reconfigurable form of parallel computing, where the DPUs (here called rDPUs) and the interconnect resources are reconfigurable, so that the pipe network is configured into a reconfigurable array. The first such reconfigurable pipe network array, the KressArray, along with a mapper and scheduler DPSS (Data Path Synthesis System) has been published in 1995. In 2000 a KressArray design space Xplorer has been implemented,

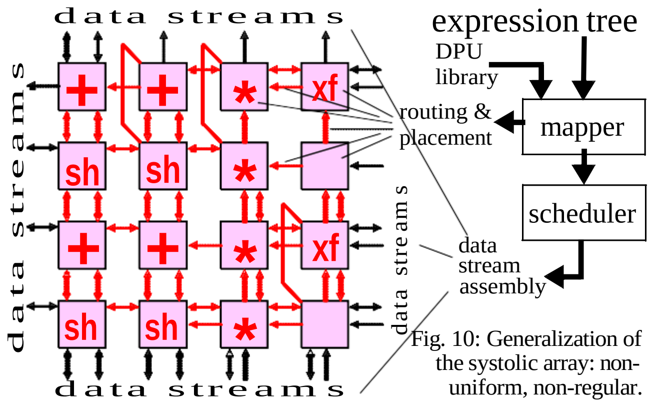


Fig. 10: Generalization of the systolic array: non-uniform, non-regular.

which supports a generically defined KressArray family covering any path width and a wide variety of inter-rDPU interconnect resources.

4.2 The key to massive parallelization

are multiple data streams being piped through a rDPU array, like through a systolic array (fig. 9), which means computing in time and space (compare fig. 6). However, for DPA synthesis no linear projection is used, but simulated annealing instead, to avoid restrictions to applications only with regular data dependencies. This generalization of the systolic array also supports inhomogenous irregular arrays (fig. 10). There are two kinds of approaches to cope with the traditional memory communication gap still widening. First, in streaming data applications like in DSP the data streams can be split up into parallel streams to be interfaced with multiple I/O ports of rDPU arrays (e. g. fig. 10). Second, artificial multiple "data streams" from/to multiple memory banks (fig. 10) can be generated by multiple data sequencers, being distributed over the rDPU array [20]. Data sequencer principles have been developed, which avoid control overhead [21]. (r)DPAs provide a much more efficient way to cope with memory communication bandwidth problems than classical concurrent computing.

New machine principles are needed for RC. Classical parallel processing relying on concurrent processes is not the way to go, since its fundamental architecture relies on a uniprocessor [21]. Generally classical parallel processing has not been successful (see fig. 5). Arrays or other ensembles of CPUs are too difficult to program, and often the run-time overhead is too high, except for a few special application areas favored by Amdahl's law. All these problems stem from the fact, that the operation of CPUs or of arrays of CPUs are control-flow-based. We need an alternative paradigm which is not control-flow-based. For details see section 7.

Stream-based ALU arrays or DPU arrays (DPU stands for Data Path Unit). Its alternative, (locally) distributed computing, uses arrays of ALUs (or other DPUs) instead of arrays of CPUs. The DPUs within such an array are interconnected form a pipe network, i. e. a network of multiple pipelines in terms of multiple DPUs (Data Path Units) without program controllers, not multiple CPUs. Tailored multiple data streams are pumped from outside through this pipe network. That's why these arrays are called "stream-based" arrays. The KressArray is an early stream-based DPU array, which is reconfigurable [6] [22] [12]. There's no CPU. There's nothing "central". It's fully distributed, with lots of different DPUs containing adders, registers, multipliers -- just what's needed for direct

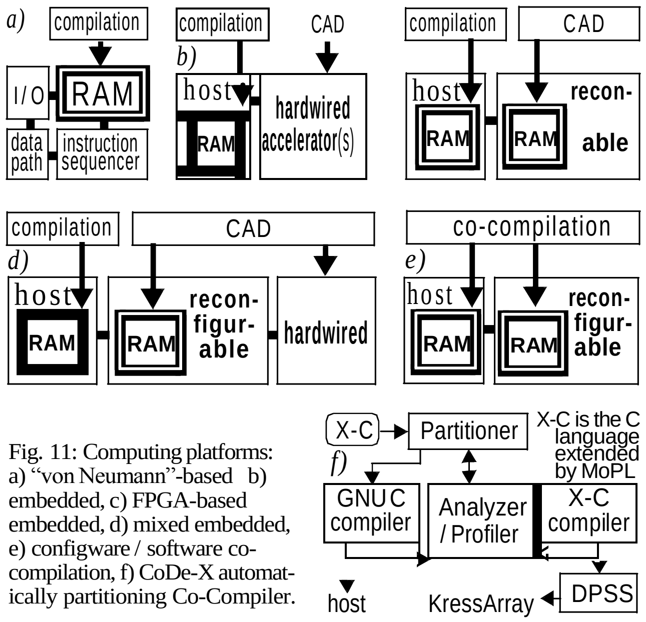
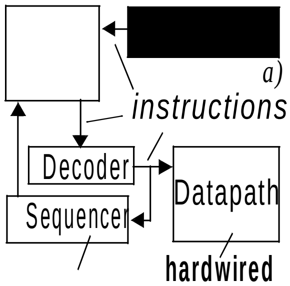


Fig. 11: Computing platforms: a) "von Neumann"-based b) embedded, c) FPGA-based embedded, d) mixed embedded, e) configware / software co-compilation, f) CoDe-X automatically partitioning Co-Compiler.



b) machine category		Computer ("v. Neumann")
machine paradigm		procedural sequencing: deterministic
driven by:		control flow
RA support		no
engine principles		instruction sequencing
state register		program counter
communication path set-up		at run time
data path	resource	single ALU
	operation	sequential

Fig. 12: Machine paradigms: a) control-procedural "von Neumann" paradigm, b) the properties of the paradigm.

mapping of the algorithm onto the architecture. As soon as the architecture is defined, the data streams needed are obtained by using a scheduling algorithm. For data stream creation see section on the memory communication gap.

On hardwired DPU array basis the BWRC [16] is developing an entire "chip in a day" design methodology by direct mapping of algorithms onto high-level, pre-characterized macros (library elements, paramete-rized blocks, like FFTs of different sizes, and a Viterbi decoder), wired together from a Simulink data-flow diagram. An automated flow goes through module generation, synthesis and layout. The easiest place to make the most profound optimizations is at the system level, where one needs to know what the implications are of the algorithmic choices you're making. Compared to full custom microprocessor design, BWRC people give up easily a factor of two or three in speed - but for gaining a factor of 100 in area efficiency. BWRC got rid of difficult problems by using relatively low clock rates. This flow gives a much more efficient way to solve the problem. Adapting the goals or Broderson's group to reconfigurable computing, where no physical design has to be done: could this end up with a FPGA-based "chip-in-one-hour" implementation?

5. Configware industry

The growing market share of embedded system applications stimulates a strong tendency to overcome separate design flows for configware development and software development. First versions of EDA tools integrating the flows into configware / software co-design are emerging from academia, and soon from FPGA or EDA vendors.

RAM-based is the xecret of success. Traditional procedural computing systems are run by software code downloaded into its RAM - basis of the extreme flexibility: The secret of success of the software industry is RAM-based. But also using recon-figurable "hardware" (FPGAs) is RAM-based: structural code (configuration code) is downloaded into the "hidden" RAM of FPGAs. Because it is non-procedural code, it cannot be compiled from software. Configware is needed instead (yet some people call it "IP cores"), as well as different compilation techniques. Hardware/software co-design turns into configware / software co-design.

A configware industry is emerging as a counterpart to software industry. Being RAM-based configware industry is taking off to repeat the success story known from software. Tsugio Makimoto has predicted this more than ten years ago [23] [24]. Like software, also configware may be downloaded over the internet, or even via wireless channels. FPGA functionality can be defined and even upgraded later at the customer's site - in contrast to the hardware it replaces: Configware use means a change of the business model - providing shorter time to market and (FPGA) product longevity. Many system-level integrated products without reconfigurability will not be competitive.

Designer productivity is the key issue. Customers aren't just looking for a million logic gates, They're looking for more than just pure logic to implement more complex systems. They're facing the problem of how to quickly cope with so many gates, how to shorten design times by predefined functional blocks to deal with more mundane or standard functions. This led vendors to focus on intellectual property and the use of code blocks to reduce design times.

IP reuse is an important issue. Clearly IP reuse and "pre-fabricated" components are factoring into the efficiency of design and use for PLDs, just as in other realms of chip development. FPGAs are going into every type of application. An FPGA, from an IP standpoint, is starting to look like an ASIC. Accordingly, the PLD vendors provide a range of libraries to enhance and facilitate the use of their products. It's hard to ignore that in today's market Altera and Xilinx own the lion's share of the PLD business.

Also the configware market is taking off for main-stream. Because FPGA-based designs become more and more complex, even entire systems on a chip, a good designer productivity and design quality cannot be obtained without good configware libraries with soft IP cores from various application areas. Currently most of the configware (reusable soft IP cores) is provided by the FPGA vendors for free as a service to their customers. But the number of independent configware houses (soft IP core vendors) and design services is growing (see section allianceCORE) and Reference Design Alliance). It may be predicted, that by the years a tendency may arise toward a growing configware market, more or less apart from the FPGA hardware market. But currently the top FPGA vendors are the key innovators and meet most of the demand in configware.

A separate EDA software market, comparable to the compiler and OS market in computers, separate from the (reconfigurable) hardware market (flexware market) is already existing, since Cadence, Mentor Graphics and Synopsys just jumped into it by closing the back end gap down to creating configware code. The battle for market shares between EDA vendors and FPGA vendors has just been started. There are still a few independent EDA software houses, not yet having been acquired by one of the major ones (see section software alliance EDA). But still less then 5% of the income of Xilinx or Altera is obtained from selling EDA software. But this may change by the time.

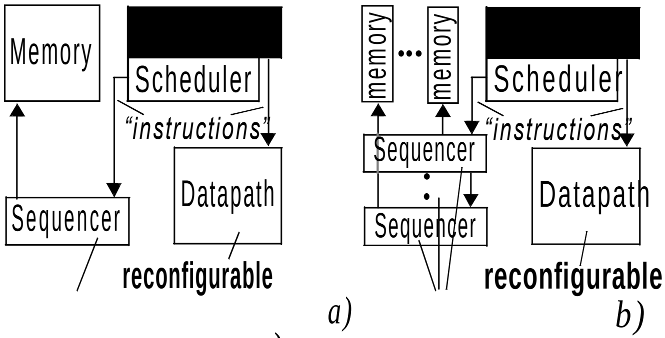
EDA is the key enabler for the customer to obtain high quality FPGA-based products with good designer productivity. Selecting FPGA architectures is not the primary key to success of customer's operations. A good FPGA architecture is useless, if it is not efficiently supported by the EDA environment(s) available. (Also see paragraph What FPGA architecture should be selected?) But EDA still often has massive software quality problems with respect to the goals to get best designer productivity and to produce highest quality designs.

Being fabless the FPGA vendors Xilinx and Altera spend most of their higher qualified manpower in EDA tool development, and IP core development, application development, and other customer support activities like related design services. So it happens, that Xilinx and Altera are more and more morphing into EDA companies. It fits into this view, that Xilinx is a fabless IC vendor and Altera has almost fabless operations (In 1999 Altera sold its MAX 5000 CPLD product line back to Cypress Semiconductor Corp., as well as its 18 percent equity interest in its Fab II wafer manufacturing facility in Round Rock, Texas). The customer's choice is, to get the development environment from the vendor of the FPGAs used, or from one of the major EDA companies (Cadence, Mentor Graphics, or Synopsys), or to assemble mixtures from both market domains.

Also the configware market is taking off for mainstream. Because FPGA-based designs become more and more complex, even entire systems on a chip, a good designer productivity and design quality cannot be obtained without good configware libraries with soft IP cores from various application areas. Currently most of the configware (reusable soft IP cores) is provided by the FPGA vendors for free as a service to their customers. But independent special configware houses (soft IP core vendors) and design services are emerging. It may be predicted, that by the years a tendency may arise toward a growing configware market, more or less apart from the FPGA hardware market. But currently the top FPGA vendors with their third party alliances are the key innovators and meet most of the demand in configware.

A separate EDA software market, comparable to the compiler and OS market in computers, separate from the (reconfigurable) hardware market (flexware market) is already existing, since Cadence, Mentor Graphics and Synopsys just jumped into it. The battle for market shares between EDA vendors and FPGA vendors has just been started.

Major changes of the EDA tools market are on the way. EDA vendors show increasing awareness of the new needs: support a business model for low cost, high volume market, good quality software requiring (almost) zero maintenance. This means a different tool quality level than still usual today. Tools have to be bullet-proof and self-supporting for a broad geographic coverage for masses of designers, with a low EDA budget. Indirect sales will be a must.



c)

machine		Xputer [26] (no transputer!)
machine paradigm		deterministic
driven by:		(**no** *dataflow* [27]) data **stream**(s)
RA support		yes
engine principles		data sequencing
state register		(multiple) data counter(s)
communication path set-up		at load time
data path	resource	array of ALUs
	operation	parallel

Fig. 13: Machine paradigms : a) data-procedural Xputer paradigm b) parallel Xputer: general model of all data-stream-based machines - counterpart of ("antimatter to") the "von Neumann" machine, c) properties of the machine paradigm.

6. Giga FPGA and soft CPU cores

Configware providers meanwhile have discovered CPUs having been developed as soft IP cores to be mapped onto an FPGA, also called FPGA CPU, or, soft CPU, like the 32 bit MicroBlaze (125 MHz, Xilinx), the Nios (16-bit instruction code, Altera), which can be configured as (8-), 16- and 32-bit data paths, and Leon, a SPARC clone by Gaisler Research. Using the usual FPGA design flow the soft CPU IP cores can be generated from VHDL or Verilog originally targeted at a hardwired implementation. The table in fig. 15 lists more soft CPU examples. Soft CPUs are also a well accepted academic research area. Some soft CPU core examples have been implemented by universities, like UCSC (already 1990), Märaldalen University, Eskilstuna, Chalmers Univ., Goeteborg, Cornell, Georgia Tech, Hiroshima City University, Michigan State, Universidad de Valladolid, Virginia Tech, Washington University, St. Louis, New Mexico Tech, UC Riverside, Tokai University.

The Giga FPGA is emerging. It can be predicted, that within a few years FPGAs with 100 million gates ore more will be available commercially. Onto a single chip of such a platform up to about a hundred soft processor cores can be mapped, leaving sufficient area to other on-board resources like RAM, register files, peripheral circuits and miscellaneous others. This could mean, that a coarse grain reconfigurable systems like those from PACT Corp. [10] can be mapped onto a (fine grain) FPGA (fig. 10). It is quite promising, that the performance disadvantage of lower clock frequency can be fixed by utilizing the better flexibility and by a much higher degree of parallelism.

Soft RC arrays will be an alternative. With FPGAs having millions of gates even soft DPU arrays (DPAs) are within reach. Reconfigurable instruction set processors The success of companies like Tensilica indicate that there is a

machine category	Computer ("v. Neumann")	Xputer [26] (no transputer!)	dataflow machine [27]
machine paradigm	procedural sequencing: deterministic		arbitration-driven indeterministic
driven by:	control flow	data stream(s)	token flow
reconfigurability support	no	yes	no
engine principles	instruction sequencing	data sequencing	arbiter decides order of execution
state register	program counter	(multiple) data counter(s)	none
communication path set-up	at run time	at load time	at run time
data path: resource	single ALU	FPGA or (r)DPU array	single ALU
data path: operation	sequential	parallel	sequential

Fig. 14: Machine paradigms: comparing Computer and Xputer.

growing market ASIPs (Application-Specific Instructionset Processors). Even more flexibility can be obtained by a soft CPU. For soft DPAs e don't need to wait for the coming Giga FPGA. Already now it is conveniently possible to map 32 MicroBlaze or 64 Nios onto a FPGA.

Excessive optimization will be needed for Giga FPGAs. If there is an infinite amount of gates available on a chip just compilation techniques can be used in front of the (gate level) configuration code generator. But for FPGAs one million gates (state of the art, or 10 million gates: may be in 2003) is far away from "infinite resources". The consequence is, that for closing the gap excessive optimization is required. This means, that leading edge designs ("bleeding edge designs") are achievable only partly with sophisticated EDA tools, so that hardware expertise is inevitable for the designer.

Contemporary FPGA architectures are not really scalable. A major problem of configware industry in competing with software industry is the fact, that no FPGA architecture is available which is fully scalable and which supports fully relocatable configuration code. The consequence is the need for re-compilation and re-debugging as soon as another FPGA type is used. It is an unanswered question, wether such a FPGA architecture is physically feasible. But it seems to be feasible in connection with a CAD tool tailored to solve this problem.

7. Compilation Techniques for RC

"von Neumann" and the classical compiler are obsolete. Today, host/accelerator(s) symbiosis is dominant and most of the platforms summarized above make use of it. Newer commercial platforms include all on a single chip, combining a core processor (ARM, or MIPS), embedded memory and reconfigurable logic. Sequential code is downloaded to the host's RAM. But accelerators are still implemented by CAD, a C compiler is only an isolated tool, and, *software / configware partitioning is still done manually* [28] [30] [31] [32] [36] [33], so that massive hardware expertise is needed to implement accelerators.

In using for recofigurable datapaths the von Neumann paradigm is falling apart. Like microprocessor usage, FPGA application is RAM-based, but by structural programming (also called "(re)configuration") instead of procedural programming. Now both, host and accelerator are RAM-based (fig. 11 c) and as such also available on the same chip: a new

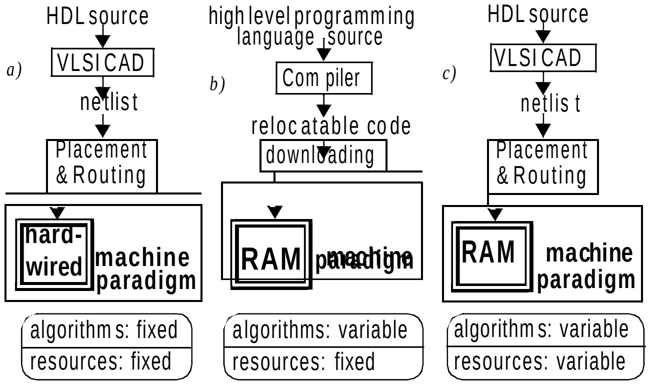


Fig. 15: Synthesis a) hardwired, b) "von Neumann", c) reconfigurable.

approach to SoC design. But the "von Neumann" paradigm does not support soft datapaths because "instruction fetch" is not done at run time, and, since most reconfigurable computing arrays do not run parallel processes, but multiple pipe networks instead. A transition from CAD (fig. 11 e) to compilation is needed, and from hardware/software co-design to configware/software co-compilation. The paper illustrates such a roadmap to reconfigurable computing, supporting the emerging trend to platform-based SoC design.

From Makimoto's wave model [6] [23] [24] point of view this reveals a huge historical gap: first wave methods "stone age" or "rubylith age" methods (fig. 15 a) for designing third wave platforms (fig. 15 c). Due to this huge gap (missing the second phase which as happened with the microprocessor: fig. 15 b): the RAM-based nature of the physical platform is mainly ignored (at this level). To close this EDA methodology gap is the dramatic challenge to EDA industry: more and more using high level programming languages, or, yet using HDLs (hardware description languages) as sources to describe design problems.

7.1 Co-Compilation

Using RAs as accelerators again changes this scenario: now implementations onto both, host and RA(s) are RAM-based, which allows turn-around times of minutes for the entire system, instead of months needed for hardwired accelerators. This means a change of market structure by migration of accelerator implementation from IC vendor to customer, demanding automatic compilation from high level programming language sources onto both, host and RA: *co-compilation* including automatic software/configware partitioning. Since compilers are based on a machine paradigm and "v. Neumann" does not support soft datapaths (because "instruction fetch" is not done at run time: we need a new paradigm (Xputer [33]) for the RA side, where the program counter is replaced by a data counter (*data sequencer* [34] [21]). Figure 14 d compares the properties of both paradigms. With multiple data sequencers a single Xputer may even handle several parallel data streams.

CoDe-X is the first such co-compilation environment having been implemented ([35] fig. 11 f), which partitions mainly by identifying loops suitable for parallelizing transformation [6] [35] [36] into code downloadable to the MoM accelerator Xputer. The MoM (Map-oriented Machine) is an Xputer architecture for data-stream-based computing [14] [17] [37].

7.2 The *Xputer* Machine Paradigm

The Xputer Machine Paradigm for soft hardware [17] [25] [38] [26] is the counterpart (fig. 13 a) of the von Neumann paradigm (fig. 12 a). But unlike "von Neumann" the Xputer

soft CPU core	architecture	platform / remarks
MicroBlaze 125 MHz 70 D-MIPS	32 bit standard RISC 32 reg. by 32 LUT RAM-based reg.	Xilinx - Virtex-II fabric (possible: up to 100 on one FPGA)
Nios	16-bit instr. set	Altera - Mercury
Nios 50 MHz	32-bit instr. set	Altera 22 D-MIPS
Nios	8 bit	Altera - Mercury
gr1040	16-bit	Altera - Mercury
gr1050	32-bit	FLEX10K30 or EPF6016
DSPuva16	i8080A	Spartan-II
Leon 25 Mhz	SPARC	Xilinx XCV800-6
ARM7 clone	ARM RISC	
uP1232 8-bit	CISC, 32 reg.	200 XC4000E CLBs
REGIS	8 bit instr. + ext. ROM	2 XILINX 3020 LCA
Reliance-1	12 bit DSP	Lattice 4 isp30256, 4 isp1016
1Popcorn-1	8 bit CISC	Altera, Lattice, Xilinx
Acorn-1		1 Flex 10K20
YARD-1A	16-b.RISC 2-opd. Instr.	old Xilinx FPGA Board
xr16	RISC integer C	SpartanXL

Fig. 15: Soft CPU IP cores currently available.

allows multiple sequencers (fig. 13 b) to support multiple data streams typical to data-stream-based computing. Figure 14 d compares the properties of both paradigms. A highly powerful generic sequencer methodology ([20] within this conference proceedings colume) to efficiently cope with memory bandwidth problems has been developed for Xputer architectures like the MoM (map-oriented machine).

Also experimental high level programming languages for that ne paradigm have been implemented, within the context of MoM (map-oriented machine) architectures [14] [17] [37]. Instead of a "control flow" sublanguage a "data stream" sublanguage like *MoPL* [39] recursively defines *data goto, data jumps, data loops, nested data loops,* and *parallel data loops*. To solve the memory bandwidth problem the Xputer paradigm is much more promising than "von Neumann".

7.3 Data Transfer and Storage Exploration (DTSE)

Currently memory bandwidth and power dissipation are the most urgent optimization problems in using design space exploration (DSE) and platform space exploration (PSE) as well as in mapping applications onto platforms [12] [13]. Caches do not help, since in data-stream-based computing there are only hardwarized loops, but no instruction streams. The processor / memory communication bandwidth gap, which spans up to 2 orders of magnitude, where new memory architectures like RAMbus or DDRAM and others bring only marginal alleviation, can be even wider in data-intensive RA use. More recently published publications on DTSE ([41] - [48]) offer a methodology for memory and communication power savings, and, loop transformations [35] [49] [36] etc. for power savings [50] [51] and speed-up - by working on data smaller local memory ([52] - [54]) instead of distant larger memory. A general architecture supporting such data locality strategies has been implemented already a decade earlier: the smart memory interface of the MoM reconfigurable architectures ([26] [55] [17] et al.), based on the generic address generator (GAG) general sequencer concept ([26] - [37] [56] [57] [58] et al.), at that time also used for a flexible and storage scheme optimization methodology [41] for concurrent multiple memory banks. It has been shown ([41] and earlier), that by using a 2-dimensional memory organization this methodology provides a rich supply of generic DTSE transforms as well as their excellent visualization. The *KressArray Xplorer* also yields solutions to the memory bandwidth problem [41], where both, data sequencers and rDPUs dedicated to the application can be mapped onto the same KressArray [9] [20]. For more details on design space explorers and data transfer and storage exploration see [2] [3] [20] [41]

8. Conclusions

The paper has given a survey on reconfigurable logic and reconfigurable computing and its R&D branches and has pointed out future trends driven by technology progress and innovations in EDA. Deep submicron allows SoC implementation, and the silicon IP business reduces entry barriers for newcomers and turns infrastructures of existing players into liability [59] [60]. The availability of reconfigurable platforms and related EDA tools

The paper has summarized the history of silicon application synthesis, which distinguishes three phases [61] [23]: hardware design (fig. 15 a), microcontroller usage (fig. b), and FPL / RA usage (fig. c). The first shift to microprocessor usage has switched the business model from structural synthesis by net-list-based CAD (fixed algorithms, no machine paradigm) to RAM-based procedural synthesis by compilation, based on a machine paradigm, which would drastically reduce the design space by guidance - the secret of success of the software industry on the procedural programming side. Also RAM-based structural programming has a huge potential for flexibility and fast turn-around and shifts product definition from hardware vendor to customer's site. It is time to switch to real compilation techniques, based on a soft machine paradigm, to go toward a dichotomy of RAM-based programming: procedural versus structural, integrating both worlds of computing. The paper has shown, that the new "soft machine" paradigm and language framework is available for such novel compilation techniques.

Many system-level integrated future products without reconfigurability will not be competitive. Instead of technology progress better architectures by reconfigurable platform usage will be the key to keep up the current innovation speed beyond the limits of silicon. It is time to revisit past decade R&D results to derive commercial solutions: at least one promising approach is available. It is time for you to get involved. Curricular innovations are urgently needed.

9. Literature

1. http://fpl.org
2. R. Hartenstein (embedded tutorial): A Decade of Research on Reconfigurable Architectures - a Visionary Retrospective; DATE 2001, Munich, March 2001
3. R. Hartenstein (invited embedded tutorial): Coarse Grain Reconfigurable Architectures; ASP-DAC'01, Yokohama, Japan, Jan 30 - Feb. 2, 2001
4. S. Guccione (keynote): Reconfigurable Computing at Xilinx; DSD 2001, Digital System Design Symposium, Warsaw, Poland, Sep 4-6, 2001
5. 13th IEEE International Workshop on Rapid System Prototyping; Darmstadt, Germany, July 1-3, 2002 http://www.rsp-workshop.org
6. R. Hartenstein: The Microprocessor is no more General Purpose (invited paper), Proc. ISIS'97, Austin, Texas, USA, Oct. 8-10, 1997.
7. A. DeHon: Reconfigurable Architectures for General Purpose Computing; report no. AITR 1586, MIT AI Lab, 1996
8. M. J. Flynn, P. Hung, K. Rudd: Deep Submicron Microprocessor Design Issues; IEEE Micro July-Aug '99
9. R. Hartenstein (invited embedded tutorial:): Reconfigurable Computing: the Roadmap to a New Business Model and its Impact on SoC Design; SBCCI 2001 - 14th Symposium on Integrated Circuits and Systems Design, Pirenopolis, DF, Brazil, September 10-15, 2001
10. http://pactcorp.com
11. J. Becker: Configurable Systems-on-Chip: Commercial and Academic Approaches; ICECS 2002, Dubrovnik, Croatia, Sept 15-18, 2002

12. U. Nageldinger et al.: KressArray Xplorer: A New CAD Environment to Optimize Reconfigurable Datapath Array Architectures; ASP-DAC, Yokohama, Japan, Jan. 25-28, 2000.
13. U. Nageldinger: Design-Space Exploration for Coarse Grained Reconfigurable Architectures; Dissertation, Universitaet Kaiserslautern, June 2001
14. R. Hartenstein, M. Herz, T. Hoffmann, U. Nageldinger: Generation of Design Suggestions for Coarse-Grain Reconfigurable Architectures; in [15]
15. R. Hartenstein, H. Grünbacher (Editors): The Roadmap to Reconfigurable computing - Proc. FPL2000, Aug. 27-30, 2000; LNCS, Springer-Verlag 2000
16. C. Chang, K. Kuusilinna, R. Broderson, G. Wright; The Biggascale Emulation Engine; FPGA 2002, 10th Int'l Symp. on Field-programmable Gate Arrays; Monterey, CA, Feb 24 - 26, 2002
17. R. Hartenstein, A. Hirschbiel, M. Weber: MoM - a partly custom-design architecture compared to standard hardware; IEEE CompEuro 1989
18. R. Hartenstein (invited paper): High-Performance Computing: über Szenen und Krisen; GI/ITG Workshop on Custom Computing, Dagstuhl, June 1996
19. R. Hartenstein (invited talk): Stream-based Arrays: Converging Design Flows for both, Reconfigurable and Hardwired .... ; IFIP International Conference on Very Large Scale Integration (VLSI-SoC 2001), December 2- 4, 2001, Montpellier, France
20. M. Herz, R. Hartenstein, M. Miranda, E. Brockmeyer, F. Catthoor: Memory Addressing Organization for Stream-based Reconfigurable Computing; ICECS 2002, Dubrovnik, Croatia, Sept 15-18, 2002
21. R. Hartenstein (invited post conference tutorial): Enabling Technologies for Reconfigurable Computing 3rd Workshop.on Enabling Technologies for System-on-Chip Development November 21, 2001, Tampere, Finland
22. R. Kress et al.: A Datapath Synthesis System for the Reconfigurable Datapath Architecture; ASP-DAC'95, Chiba, Japan, Aug. 29 - Sept. 1, 1995
23. T. Makimoto: The Rising Wave of Field-Programmability; in: [15]
24. T. Makimoto, D. Manners: Living with the Chip; Chapman & Hall, 1993
25. R. Hartenstein et al.: A Novel Paradigm of Parallel Computation and its Use to Implement Simple High Performance Hardware; InfoJapan'90, 30th Anniversary of the Computer Society of Japan, Tokyo, Japan, 1990.
26. R. Hartenstein, A. Hirschbiel, K. Schmidt, M. Weber: A Novel Paradigm of Parallel Computation and its Use to Implement Simple High-Performance-HW; Future Generation Computer Systems 7, 91/92, North Holland - invited reprint of [25]
27. D. Gajski et al.: A second opinion on dataflow machines; Computer, Feb 1982
28. T. J. Callahan and J. Wawrzynek: Instruction-Level Parallelism for Reconfigurable Computing; in [29] pp. 248-257.
29. R. Hartenstein, A. Keevallik (Editors): Proc. FPL'98, Tallinn, Estonia, Aug. 31- Sept. 3, 1998, LNCS, Springer Verlag, 1998
30. M. Budiu and S. C. Goldstein: Fast Compilation for Pipelined Reconfigurable Fabrics; Proc. FPGA'99, Monterey, Feb. 1999, pp. 135-143.
31. M. Weinhardt, W. Luk: Pipeline Vectorization for Reconfigurable Systems; Proc. IEEE FCCM, April 1999
32. M. Gokhale, J. Stone: NAPA C: Compiling for a hybrid RISC / FPGA architecture; Proc. IEEE FCCM April 1998
33. J. Becker et al.: A General Approach in System Design Integrating Reconfigurable Accelerators; Proc. IEEE ISIS'96; Austin, TX, Oct. 9-11, 1996
34. M. Herz, et al.: A Novel Sequencer Hardware for Application Specific Computing; Proc. ASAP'97, Zurich, Switzerland, July 14-16, 1997
35. J. Becker: A Partitioning Compiler for Computers with Xputer-based Accelerators; Ph. D. dissertation, Kaiserslautern University, 1997.
36. K. Schmidt: A Program Partitioning, Restructuring, and Mapping Method for Xputers; Ph.D. Thesis, University of Kaiserslautern 1994
37. R. Hartenstein, A. Hirschbiel, M. Weber: MOM - Map Oriented Machine; in: E. Chiricozzi, A. D'Amico: Parallel Processing and Applications, North-Holland, 1988
38. R. Hartenstein et. al.: A Novel ASIC Design Approach Based on a New Machine Paradigm; IEEE J.SSC, Volume 26, No. 7, July 1991.
39. A. Ast, J. Becker, R. Hartenstein, R. Kress, H. Reinig, K. Schmidt: Data-procedural Languages for FPL-based Machines; in [40]
40. R. Hartenstein, M. Servit (Editors): Proc. FPL'94, Prague, Czech Republic, Sept. 7-10, 1994, LNCS, Springer Verlag, 1994
41. M. Herz: High Performance Memory Communication Architectures for Coarse-Grained Reconfigurable Computing Architectures; Ph. D. Dissertation, Universitaet Kaiserslautern, January 2001
42. F.Cathoor et al.: Custom Memory Management Methodology; Kluwer, 1998
43. N. D. Zervas et al.:Data-Reuse Exploration for Low-Power Realization of Multimedia Applications on Embedded Cores; PATMOS'99, Kos Island, Greece, Oct 1999
44. D. Soudris et al.: Data-Reuse and Parallel Embedded Architectures for Low-Power Real-Time Multimedia Applications; PATMOS'2000, see: [45]
45. D. Soudris, P. Pirsch, E. Barke (Editors): Proc. PATMOS 2000; Göttingen, Germany Sept. 13 - 15, 2000; LNCS, Springer Verlag, 2000
46. J. Kin et al.: Power Efficient Media Processor Design Space Exploration; Proc. DAC'99, New Orleans, June 21-25, 1999
47. S. Kougia et al.: Analytical Exploration of Power Efficient Data-Reuse Transformations on Multimedia Applications; ICASSP'2001, Salt Lake City, May 2001
48. S. Wuytak et al.: Formalized Methodology for Data Reuse Exploration for Low Power Hierarchical Memory Mappings; IEEE Trans. on VLSI Systems, Dec. 1998
49. D. Kulkarni, M. Stumm: Loop and Data Transformations: A tutorial; CSRI-337, Computer Systems Research Institute, University of Toronto, June 1993
50. V. Tiwari et al.: Power Analysis of Embedded Software: A First Step Towards Software Power Minimization; IEEE Trans. on VLSI Systems, Dec. 1994
51. G. Sinevriotis et al.: Power Analysis of the ARM 7 Embedded Microprocessor; PATMOS'99, Kos Island, Greece, Oct 1999
52. A. Vandecapelle et al.: Global Multimedia Design Exploration using Accurate Memory organization Feedback; Proc. DAC 1999
53. T. Omnès et al: Dynamic Algorithms for Minimizing Memory Bandwidth in High throughput Telecom and Multimedia; in: Techniques de Parallelization Automatique, TSI, Éditions Hermès, 1999
54. S. Wuytack et al: Minimizing the required Memory Bandwidth in VLSI System Realizations; IEEE Trans. VLSI Systems, Dec. 1999
55. R.W. Hartenstein, A.G. Hirschbiel, M. Weber: A Flexible Architecture for Image Processing; Microprocessing and Microprogramming, vol 21, pp 65-72, 1987
56. R. Hartenstein, A. Hirschbiel, K. Schmidt, M. Weber: A Novel ASIC Design Approach based on a New Machine Paradigm; Proc. ESSCIRC`90, Grenoble, France
57. R. Hartenstein, A. Hirschbiel, M. Riedmüller, K. Schmidt, M.Weber: A High Performance Machine Paradigm Based on Auto-Sequencing Data Memory; HICSS-24, Hawaii Int'l. Conference on System Sciences, Koloa, Hawaii, 1991
58. Reiner W. Hartenstein, Helmut Reinig: Novel Sequencer Hardware for High-Speed Signal Processing; Workshop on Design Methodologies for Microelectronics, Smolenice Castle, Slovakia, September 1995
59. T. Kean (invited keynote): It's FPL, Jim - but not as we know it! Market Opportunities for the new Commercial Architectures; in [15]
60. R. Hartenstein (invited keynote): Reconfigurable Computing: a New Business Model - and its Impact on SoC Design; DSD'2001 Warsaw, Poland, Sept 4 - 6, 2001
61. N. Tredennick: Technology and Business: Forces Driving Microprocessor Evolution; Proc. IEEE 83, 12 (Dec. 1995)