The Digital Divide of Computing

Reiner Hartenstein
TU Kaiserslautern
http://hartenstein.de

ABSTRACT

This presentation urges for creating more awareness of the impact of configware engineering onto embedded system development and examines the requirements of overdue CSE curricular upgrades. Because of the impact of reconfigurable computing, configware engineering has proceeded from niche to mainstream. Morphware has become an essential and indispensable ingredient in SoC (System on a Chip) design and beyond. It turns embedded system design from hardware / software co-design into configware / software co-design [1] [2] [3] [4]. This hot development, supported by the fastest growing segment of the semiconductor market [5], provides morphware [6] [7] as an alternative RAM-based "programmable" (more precisely called: "reconfigurable") platform for parallelism avoiding the limitations of classical high performance computation [8] [9] [10] caused by the von Neumann paradigm [11] [12] [13]. The digital divide of computing determines who is qualified to take off toward new horizons in high performance computing, and, who is not. Currently the typical CS graduate is not.

Categories and Subject Descriptors

C.4 **[Performance of Systems]**

General Terms

Performance, Design, Standardization

Keywords

Performance, Morphware, SoC, reconfigurable computing

1. A HUGE DISASTER

Meanwhile it is widely accepted, that morphware is a new computing paradigm [14]. However, a major problem for further progress is the lack of qualified experts. The hardware / software chasm in professional practice and in education causes a damage amounting to billions of EURO each year worldwide. It is a main reason of the productivity gap in embedded system design. This traditional hardware / software chasm is deepening into the software / configware chasm. This digital divide of computing is driving the entire IT area into a huge disaster.

The amount of code for embedded systems, to be implemented by programmers, doubles every 10 months [15] [16] and will reach 90% of all code being written by about the year 2010 [16]. Most programmers are not qualified for this task, preparing a disaster for the IT job market of the near future. Currently a typical CS graduate with von-Neumann-only mentality does not have the skills needed for hardware / configware / software partitioning decisions, nor the algorithmic cleverness needed to migrate an application from software onto morphware. The highly powerful, but also provocative new mind set of configware, hidden behind reconfigurable computing methodology, currently is not accessible by programmers or CS an CSE graduates with a traditional background.

So we have a digital divide between those being able to cope with configware and to utilize their benefits, and those, who can't. Morphware provides the enabling fundamentals, methodologies, and technologies to cope with this crisis. But widely spread awareness is still missing. Often the digital divide will decide, who will get the job. This problem can be solved only by a fundamental EECS curricular revolution. The results of a decade of R&D are available for education and commercialization: to cope with the current SoC design crisis by a transition from hardware / software co-design to platform-based SoC design by configware / software co-compilation.

Limitations of von-Neumann-based computing

The future of the microprocessor promises only marginal improvements for performance, low power, and area efficiency. Multi-threading and pipelined execution units yield only marginal benefit for the price of substantial overhead [8]. Power dissipation is rapidly going worse. The intel Itanium 2 dissipates 130 Watts at 1,3 Volts [17]. Contemporary High Performance Computing needs about 100W per gigaFLOPS [83]. The von Neumann bottleneck still is the dominating limitation [9] [10] [13] [18] [19]. We need a new computing paradigm for morphware accelerators. Along with a good co-compiler adding such accelerators to a microprocessor may turn the PC into a PS (personal supercomputer). A highly promising alternative is the microprocessor interfaced to a suitable coarse grain array, maybe for converting a PC into a PS (personal supercomputer). But such a PS will be accepted by the market only, when it comes along with a good co-compiler, the feasibility of which has been demonstrated [20] [21] [22].

2. RECONFIGURABLE COMPUTING

Supporting only fine-grained reconfigurability of roughly single bit wide configurable logic blocks (CLBs) the mapping tools for FPGAs are mainly based on gate level methods - similar to CAD for hardwired logic, so that hardware experts are needed. However, from a decade of world-wide research on Reconfigurable Computing another breed of reconfigurable platforms is an emerging competitor to FPGAs [7] [23]: *Coarse Grain Morphware*. In contrast to FPGAs, the reconfigurable computing scene usually works with arrays (rDPAs) of coarse-grained reconfigurable data path units (rDPUs) with drastically reduced reconfigurability overhead: to directly configure high level parallelism similar to instruction-level parallelism.

In contrast to FPGAs, coarse-grained morphware implemented in full-custom layout style may easily reach the area-efficiency of full-custom hardwired computational data paths [24] [25]. Because the number of CFBs (configurable function blocks) is by orders of magnitude smaller than that of CLBs or rDPUs in FPGAs, mapping takes only minutes or less instead of hours. Since computational data paths have regular structure potential, full custom designs of *Reconfigurable Data Path Units* (rDPUs) are drastically more area-efficient. Coarse-grained architectures provide operator level CFBs, and very area-efficient data path routing switches. A major benefit is massive reduction of configuration memory and configuration time, and drastic complexity reduction of the compilation task, here by P&R (placement and routing). But the classical machine paradigm like von Neumann does not support soft data paths because “instruction fetch” (her more precisely called: “reconfiguration”) is not done at run time.

Reconfigurable Computing vs Parallel Processing

Classical parallelism by concurrent computing has a number of disadvantages over the parallelism by anti machines having no von Neumann bottleneck, what is discussed elsewhere [9] [18] [26] [27]. In Parallel Computing, unfortunately, the scaling of application performance often cannot match the peak speed the resource platforms seem to provide, and the programming burden for these machines remains heavy. Within Concurrent Computing systems, however, the instruction fetch and set-up of all related communication paths happens *during run time*, which we *do not call reconfiguration*. The main difference with respect to performance is the amount of switching activity at run time, which is low for reconfigurable systems and high for the instruction-stream-driven such classical parallel computing.

Depending on the application and the architecture, massively parallel concurrent systems may heavily suffer from communication congestion at run time. Within Reconfigurable Computing systems, however, the “instruction fetch” (i. e. set-up of all computational resources and the set-up of all related communication paths) happens *before run time*, what we call *reconfiguration,* because it changes the effective structure of data paths and similar resources. Compared to classical parallelism reconfigurable computing can be much more efficient for most application areas. Based on free-form pipe networks, driven by multiple data streams there are hardly any memory bandwidth bottlenecks, nor organizational overhead at run time. This will be discussed by the following paragraphs on data-stream-based computing.

3. DATA-STREAM-BASED COMPUTING

The instruction-stream-based von Neumann mind set does not support configware compilation - in contrast to its rapidly upcoming counterpart, the data-stream-based basic machine model, the application of which has been popularized recently by a number of projects in reconfigurable computing ([28] - [33] and others). Already during the 80ies the systolic array R&D scene has developed a methodology of data-stream-based computing. In contrast to von-Neumann-based operation, which is driven by an instruction stream (compiled from *software* sources), the operation of super-systolic arrays and other coarse grain morphware usually is driven by data streams (compiled from *flowware* sources). A flowware source defines [34] [35] [36], which data item has to hit which rDPA port at which time. From this point of view reconfigurable computing is a kind of innovating revival of the systolic array methodology [37] [38] [39]. Because of using the wrong synthesis method [13] the systolic array, an early flowware-based paradigm, got stuck in a niche for long - until the super systolic array [40] made it viable for being a morphware platform.

The Anti Machine

Data-stream-based computing is the direct counterpart of von-Neumann-based concurrent computing Data-stream-based computing, along with its anti machine paradigm, disrupts the fundamentals of computing science. Not only for economic reasons it is overdue to develop a widely spread awareness on this revolutionary development. In contrast to the von Neumann paradigm the anti machine’s sequencer (*data counter*) has moved to the memory (as part of an *asM* [25] [41] [42] [44], an auto-sequencing memory bank), whereas the DPU of the anti machine has no sequencer. The anti machine paradigm also supports multiple data streams by multiple asMs providing multiple data counters. That’s why the anti machine has no von Neumann bottleneck.

The enabling technologies for *anti machine* architectures and their implementations are available from many research sources ([45] - [51]). The anti machine paradigm is useful for both, morphware-based machines and hardwired machines ([52] etc.). The anti machine should not replace von Neumann. We need both machine paradigms. We need morphware to strengthen the declining von Neumann paradigm. The anti machine is not a *dataflow machine* [53]. Data-stream-oriented anti machine platforms are also available commercially, like the XPP (Xtreme Processing Platform [54] [55] [56] [57]) from PACT [58], coming along with compilation tools [59]. The term *dataflow machine* cannot be used for the *anti machine*, because it had been established by an old research area (dead, meanwhile) having worked on arbitration-driven machine architectures.

4. SOC DESIGN ADOPTING CS MENTALITY

In EDA (electronic design automation) there is a trend toward higher abstraction levels for design entry, for instance with languages like system-C [60] [61] [62]. Even math formula are investigated for use as EDA design entry [63] [64]. Already HDLs (Hardware Descriptive Languages) like VHDL (an Ada dialect), Verilog (a C dialect), or others, are languages at CS-like higher abstraction levels, and should be taught also to CS students. SoC design for embedded systems rapidly adopts CS mentality [65]. The amount of program code implemented for embedded systems doubles every 10 months. There is a trend to convey Co-design of embedded computing systems from the domain of hardware expertise over to CS methodologies. To cope with this challenge to CS curricula the new anti machine paradigm and new compilation methods are needed.

This is hardly possible without moving to higher abstraction levels. Because it focuses the design space like known for software from the *von Neumann paradigm*, a second machine paradigm is needed as a simple guideline to implement flowware (and configware). The emerging transition from HDLs to configware sources of higher abstraction levels, like System-C and others, encourages to go from complex design flows to compilation techniques which are supported by an alternative simple machine paradigm model. Also dataflow languages having come along with the indeterministic dataflow machine paradigm [10] could also be candidate sources for the anti machine.

Co-Compilation

Flowware languages have close similarities to traditional imperatively procedural languages like C or Pascal, but are more easy to learn [2] [10] [66]. In addition to software compilers we need configware compilation. A first step in introducing morphware-oriented compilation techniques in application development for embedded systems is the replacement of EDA by compilation also for the morphware part. We need *two programming sources: configware* to program the resources, *and, flowware* to program the data streams running through the resources. An early implementation is the DPSS (Data Path Synthesis System [40]), part of a software / configware / flowware co-compiler [20] [21] [22], where partitioning is based on *loop transformations* ([67] [68] [69] [70] and others). A newer version of DPSS includes *KressArray Xplorer*, a design space explorer to optimize DPU and rDPA architectures [72] [73] [74].

Separate compilation of software and configware gives only a limited support to reach the goal of good designer productivity. Especially to introduce *software / configware / flowware co-design* to CS professionals and CS curricula we need *co-compilation techniques* to support application development at high abstraction levels - as the new enabling technology for highest performance computing, e. g. by the PS (personal supercomputer): a symbiosis of microprocessor and anti machine. To introduce the new business model to cope with the current accelerator design crisis a transition from CAD to compilation is needed, and from hardware/software co-design to configware/software co-compilation.

5. THE COMING DICHOTOMY OF PARADIGMS

The secret of success of CS and of the software industry is RAM-based, because different software can easily be downloaded to the customer's side at any time. Stimulated by the impact of the emerging methodology around reconfigurable computing on the classical mind set of control-flow-driven computing we are heading toward a dichotomy of computing sciences - and also toward a dichotomy within IT industry: software industry and configware industry. We are already beginning to practice a business model, where configware is downloaded to the morphware resident at the customer's site. But this paradigm switch is still widely ignored: Configware industry did not yet really repeat the RAM-based success story of the software industry. There is not yet a configware industry, since mapping applications onto morphware is still mainly practiced like a kind of hardware synthesis method, but not really by compilation. This is an employee qualification problem. It is time to teach the enabling methodologies being available already for quite a time.

Morphware provides the enabling fundamentals to cope with this crisis. It is time to bridge the hardware / software chasm. We need Mead-&-Conway-like rush [84]. We are already on the road. Scientific Computing more and more uses Morphware. The international HPC conference IPDPS is coming along with the rapidly growing Reconfigurable Architectures Workshop (RAW [86]). The number of attendees from the HPC scenes coming to conferences like FPL [87] and RAW is rapidly increasing. Special interest groups of professional organizations are changing their scope, like e. g. PARS [88] [89] [90] and tutorials have been held ([71], [75], [91] - [93]).

New taxonomy needed

The growth rate of algorithmic complexity [49] is higher than that of Moore's law (1), whereas the growth rate of microprocessor integration density (2) is far behind Moore's law). This requires more algorithmic cleverness than currently available from CS graduates' qualification.

To support the algorithmic cleverness required for a good morphware-based designer productivity and quality we need an all-embracing comparative taxonomy of architectures and algorithms, covering classical parallel computing, supercomputing, and reconfigurable computing. A new taxonomy has to come along with a consolidation of terminology. Reconfigurable versus parallel computing is also a very important issue for terminology - to avoid confusion. Unfortunately the distinction between parallel and reconfigurable computing is blurred by some projects labelled "reconfigurable", but which, in fact, are dealing with classical parallel computing on a single chip.

6. CONCLUSIONS

There is sufficient evidence that morphware is breaking through as a new computing paradigm ([100] - [108]). Breaking away from the current mind set requires more than traditional technology development and infusion. It requires managerial commitment to a long-term plan to explore new thinking [85]. Morphware has just achieved its break-through as a second class of RAM-based programmable data processing platforms - counterpart of the RAM-based von Neumann paradigm. Morphware combines very high flexibility by programmability, with the performance and efficiency of hardwired accelerators.

Already HDLs like VHDL (which is an Ada dialect), Verilog (a C dialect), or others, are languages at CS-like higher abstraction levels, and should be taught also to CS students. We need more analysts and curriculum innovators. But most current work on reconfigurable systems is specialized and is not motivated by long term aspects - wearing blinders limiting the view to particular applications, architectures, or tools. The long term view, however, shows a heavy impact of reconfigurable computing onto the intellectual infrastructures of CS and CSE. This chapter has drafted a road map for upgrading CS and CSE curricula and for bridging the gap between procedural and structural mentality. The impact of morphware on CS helps to achieve this by evolution, rather than by revolution. You all should be evangelists for the diffusion of the visions needed to go this road out of the current crisis. It is time to bridge the configware / software chasm. We need a Mead-&-Conway-like rush [84].

7. REFERENCES

[1] J. Becker, R. Hartenstein (sollicited paper): Configware and Morphware going Mainstream; Journal of System Architecture, October 2003

[2] Reiner Hartenstein (invited chapter): Morphware; in: A. Zomaya (editor): Handbook of Innovative Computing; Springer Verlag Heidelberg / New York - to appear in 2004

[3] Reiner Hartenstein (opening keynote): Are we ready for the Breakthrough ?; 10th Reconfigurable Architectures Workshop 2003 (RAW 2003), Nice, France, April 22, 2003

[4] Reiner Hartenstein (keynote address): Software or Configware? About the Digital Divide of Parallel Computing; 18th International Parallel and Distributed Processing Symposium (IPDPS), April 26–April 30, 2004, Santa Fe, New Mexico, USA

[5] B. Lewis, Gartner Dataquest, October 28, 2002

[6] Reiner 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 (together with: SBMICRO 2001, Int'l Conf. on Microelectronics and Packaging and: SBAC-PAD 2001, 13th Symp. on Computer Architecture and High Performance Computing); Pirenopolis, Brazil, September 10-15, 2001

[7] Reiner Hartenstein (invited embedded tutorial): A Decade of Reconfigurable Computing: A Visionary Retrospective; DATE 2001 - Design, Automation and Test in Europe, Conference & Exhibition 13-16 March, 2001. ICM/Neue Messe, Munich, Germany

[8] G. Koch et al.: The Universal Bus Considered Harmful; Proc. 1st EUROMICRO Symposium on the microarchitecture of computing systems; Nice, France, 1975; North Holland, 1975

[9] Arvind et al.: A critique of Multiprocessing the von Neumann Style; Proc. ISCA 1983

[10] J. Backus: Can programming be liberated from the von Neumann style? A functional style and its algebra of programs; Communications of the ACM, August 1978, 20(8):613-641.

[11] A. Burks, H. Goldstein, J. von Neumann: Preliminary discussion of the logical design of an electronic computing instrument; US Army Ordnance Department Report 1946.

[12] Goldstein, H., von Neumann, J., and Burks, A.: Report on the mathematical and logical aspects of an electronic computing instrument; Princeton Institute of advanced study, 1947.

[13] Reiner Hartenstein (invited paper - award: honorable mention): The Microprocessor is no more General Purpose: why Future Reconfigurable Platforms will win; Proc. International Conference on Innovative Systems in Silicon, ISIS'97, Austin, Texas, USA, October 8-10, 1997

[14] Reiner Hartenstein (keynote address): The impact of Morphware on Parallel Processing; 12th Euromicro Conference on Parallel, Distributed and Network based Processing (PDP04); February, 11-13, 2004. A Coruña, Spain,

[15] F. Rammig: Eingebettete Systeme; 10th anniversary workshop, Fraunhofer EAS Dresden, April 2002

[16] N. N., Department of Trade an Industry (DTI), London, UK, 2001

[17] M. Sauer: Issues in Concept Development for Embedded Wireless SoCs; GI/ITG FG AH - Zielplan-Workshop; Frankfurt / Main, Germany, Sept 22, 2003

[18] G. Bell (keynote): All the Chips Outside: The Architecture Challenge; Proc. ISCA 2000

[19] J. Hennessy: ISCA25: Looking Backward, Looking Forward; Proc. ISCA 1999

[20] J. Becker et al.: Parallelization in Co-Compilation for Configurable Accelerators; Proc. ASP-DAC´98

[21] J. Becker: A Partitioning Compiler for Computers with Xputer-based Accelerators, Ph. D. Dissertation, University of Kaiserslautern 1997 - downloadable from [22]

[22] http://xputers.informatik.uni-kl.de/papers/publications/BeckerDiss.pdf

[23] Reiner Hartenstein (invited embedded tutorial): Coarse Grain Reconfigurable Architecture; ASP-DAC 2001 - Asia and South Pacific Design Automation Conference 2001, January 30 - February 2, 2001, Conference Center Pacifico, Yokohama, Japan

[24] Reiner Hartenstein (invited paper): Trends in Reconfigurable Logic and Reconfigurable Computing; 9th IEEE International Conference on Electronics, Circuits and Systems - ICECS 2002, September 15-18, 2002, Dubrovnik, Croatia

[25] M. Herz et al. (invited paper): Memory organization for Data-Stream-based Reconfigurable Computing; Proc. IEEE - ICECS 2002

[26] C. Chang, K. Kuusilinna, R. Broderson: The Biggascale Emulation Engine; FPGA 2002

[27] M. Weber et al.: MOM - Map Oriented Machine; in: E. Chiricozzi, A. D'Amico (editors): Parallel Processing and Applications, North-Holland, 1988

[28] J. Frigo, et al.: Evaluation of the streams-C C-to-FPGA compiler: an applications perspective; FPGA 2001

[29] T. J. Callahan: Instruction-Level Parallelism for Reconfigurable Computing; FPL'98

[30] E. Caspi, et al.: Extended version of: Stream Computations Organized for Reconfigurable Execution (SCORE): FPL '2000

[31] T. Callahan: Adapting Software Pipelining for Reconfigurable Computing; CASES 2000

[32] H. Kwok-Hay So, BEE: A Reconfigurable Emulation Engine for Digital Signal Processing Hardware; M.S. thesis, UC Berkeley, 2000

[33] B. Mei et al.: Exploiting Loop-Level parallelism on Coarse-Grained Reconfigurable Architectures Using Modulo Scheduling; DATE 2003

[34] Reiner Hartenstein (invited paper): Data-Stream-Based Computing: Models and Architectural Resources; International Conference on Microelectronics, Devices and Materials (MIDEM 2003), Castle of Ptuj, Slovenia, Oct.1-3, 2003

[35] Reiner Hartenstein (invited paper): Datastream-based Reconfigurable Computing; Dresdner Arbeitstagung Schaltungs- und Systementwurf (Workshop on Circuit and Systems Design - DASS´2003), in conjunction with the Workshop System Design Automation (SDA´2003); Dresden, Germany, May, 8 - 9, 2003

[36] Reiner Hartenstein (invited paper): Data-Stream-based Computing and Morphware; Joint 33rd Speedup and 19th PARS Workshop (Speedup / PARS 2003), Basel, Switzerland, March 19 - 21, 2003

[37] N. Petkov: Systolic Parallel Processing; North-Holland; 1992

[38] M. Foster, H. Kung: Design of Special-Purpose VLSI Chips: Example and Opinions. ISCA 1980

[39] H. T. Kung: Why Systolic Architectures? IEEE Computer 15(1): 37-46 (1982)

[40] R. Kress et al.: A Data path Synthesis System for the Reconfigurable Data path Architecture; ASP-DAC'95

[41] M. Herz et al.: A Novel Sequencer Hardware for Application-specific Computing; Proc. ASAP '97

[42] M. Herz: High Performance Memory Communication Architectures for Coarse-grained Reconfigurable Computing Systems; Ph. D. thesis, Kaiserslautern, 2001 -- downloadable from: [43]

[43] http://xputers.informatik.uni-kl.de/papers/publications/HerzDiss.html

[44] R. Hartenstein, Hirschbiel, K. Schmidt, M. Weber (2nd Best Paper Award): A High Performance Machine Paradigm Based on Auto-Sequencing Data Memory; HICSS-24, Hawaii Int'l. Conf. on System Sciences, Koloa, Hawaii, 1991

[45] F. Catthoor et al.: Data Access and Storage Management for Embedded Programmable Processors; Kluwer, 2002

[46] F. Catthoor et al.: Custom Memory Management Methodology Exploration of Memory Organization for Embedded Multimedia Systems Design; Kluwer, 1998

[47] K. Schmidt et. al.: A Novel ASIC Design Approach Based on a New Machine Paradigm; J. SSC 1991 - invited reprint fr. Proc. ESSCIRC 1990

[48] W. Nebel et al.: PISA, a CAD Package and Special Hardware for Pixel-oriented Layout Analysis; ICCAD 1984

[49] R. Hartenstein et al.: A Novel Paradigm of Parallel Computation and its Use to Implement Simple High Performance Hardware; Proc. InfoJapan'90 (Int'l Conf. commemorating the 30th Anniversary of the Computer Society of Japan), Tokyo, Japan, 1990

[50] Reiner Hartenstein, Hirschbiel, K. Schmidt, M. Weber (invited reprint of [49]): A Novel Paradigm of Parallel Computation and its Use to Implement Simple High Performance Hardware; Future Generation Computer Systems 7 91/92, p. 181-198, North Holland Publishing Co. Amsterdam / New York;

[51] H. Reinig et al.: Novel Sequencer Hardware for High-Speed Signal Processing; Proc. Design Methodologies for Microelectronics, Smolenice, Slovakia, Sept.1995

[52] C. Chang et al: The Biggascale Emulation Engine (Bee); summer retreat 2001, UC Berkeley

[53] D. Gajski et al.: A second opinion on dataflow machines; Computer, Febr. 1982

[54] V. Baumgarten, et al.: PACT XPP - A Self-Reconfigurable Data Processing Architecture; ERSA 2001

[55] J. Becker, A. Thomas, M. Vorbach, G. Ehlers: Dynamically Reconfigurable Systems-on-Chip: A Core-based Industrial/Academic SoC Synthesis Project; IEEE Workshop Heterogeneous Reconfigurable SoC; April 2002, Hamburg, Germany

[56] J. Becker, M. Vorbach: An Industrial/Academic Configurable System-on-Chip Project (CSoC): Coarse.grain XPP/Leon-based Architecture Integration; DATE 2003

[57] V. Baumgarten, G. Ehlers, F. May, A. Nückel, M. Vorbach, M. Weinhardt: PACT XPP - A Self-Reconfigurable Data Processing Architecture; The Journal of Supercomputing, vol. 26, no. 2, Sept. 2003, pp. 167-184, Kluwer Academic Publishers

[58] http://pactcorp.com

[59] J. Cardoso, M. Weinhardt: From C Programs to the Configure-Execute Model; DATE 2003

[60] T. Grötker et al.: System Design with System-C; Kluwer, 2002

[61] http://www.synopsys.com/products/cocentric_systemC/cocentric_systemC_ds.html

[62] http://www.systemc.org/

[63] Reiner Hartenstein, R. Jacobi, M. Ayala-Rincon, C. Llanos: Using Rewriting-Logic Representation for Functional Verification in Data-Stream-Based Reconfigurable Computing; ISCI Forum on Specification and Design Languages (FDL'03), September 23 - 26, 2003 – Frankfurt, Germany

[64] C. Llanos, M. Ayala-Rincón, R. B. Nogueira, R. P. Jacobi, R. Hartenstein: Modeling Dynamically Reconfigurable Systems via Rewriting-Logic; Dagstuhl Seminar Nº 03301, Dynamically Reconfigurable Architectures; Dagstuhl, Germany, 20. 07.-25. 07. 2003,

[65] http://public.itrs.net/Files/2002Update/2002Update.htm

[66] A. Ast, et al.: Data-procedural Languages for FPL-based Machines; FPL'94

[67] J. Becker, K. Schmidt: Automatic Parallelism Exploitation for FPL-based Accelerators; Hawaii Int'l. Conf. on System Sciences (HICSS'98), Big Island, Hawaii,1998

[68] L. Lamport: The Parallel Execution of Do-Loops; C. ACM 17,2, Febr. 1974

[69] D. Loveman: Program Improvement by Source-to-Source Transformation; J. ACM 24,1, Jan. 1977

[70] J. Allen, K. Kennedy: Automatic Loop Interchange; Proc. ACM SIGPLAN'84, Symp. on Compiler Construction, Montreal, SIGPLAN Notices 19, 6, June 1984

[71] Reiner Hartenstein (invited full day course): Enabling Technologies for Reconfigurable Computing and Software / Configware Co-Design; CNRS internal workshop, ENST, Paris, July 8, 2002 -

[72] U. Nageldinger et al.: Generation of Design Suggestions for Coarse-Grain Reconfigurable Architectures; FPL 2000

[73] U. Nageldinger: Coarse-grained Reconfigurable Architectures Design Space Exploration; Dissertation, 2001 -- downloadable from [74]

[74] http://xputers.informatik.uni-kl.de/papers/publications/NageldingerDiss.html

[75] Reiner Hartenstein (invited full day post conference tutorial): Enabling Technologies for Reconfigurable Computing; 3rd Workshop on Enabling Technologies for System-on-Chip Development, November 21, 2001, Tampere, Finland

[76] Reiner Hartenstein M. Riedmuller, K. Schmitt, M. Weber (invited reprint): A Novel Asic Design Approach Based on a New Machine Paradigm; IEEE Journal of Solid State Circuits, July 1991, invited reprint from Proc. ESSCIRC 1990, Geneva, Switzerland

[77] Reiner Hartenstein, A.Hirschbiel, M. Riedmuller, K. Schmidt, M.Weber (best paper and best presentation award): Automatic Synthesis of Cheap Hardware Accelerators for Signal Processing and Image Preprocessing; 12th DAGM-Symposium Mustererkennung (Pattern Recognition), Oberkochen-Aalen, Germany, 1990

[78] Ulrich Nageldinger: Coarse-grained Reconfigurable Architectures Design Space Exploration; Dissertation, TU Kaiserslautern, 2001

[79] Michael Herz: High Performance Memory Communication Architectures for Coarse-grained Reconfigurable Computing Systems; Dissertation, TU Kaiserslautern, 2001

[80] Juergen Becker: A Partitioning Compiler for Computers with Xputer-based Accelerators, Ph. D. Dissertation, TU Kaiserslautern, 1997

[81] Rainer Kress: A Fast Reconfigurable ALU for Xputers, Ph. D. Dissertation, TU Kaiserslautern, 1996,

[82] Karin Schmidt: A Program Partitioning, Restructuring, and Mapping Method for Xputers, Ph. D. Dissertation, TU Kaiserslautern, 1994, Shaker Verlag, ISBN: 3-8265-0495-X

[83] I. Jones: DARPA funded Directions in Embedded Computing; THALES Reconfigurable Computing Workshop, Orsay, France, Sept. 2003

[84] http://xputers.informatik.uni-kl.de/staff/hartenstein/eishistory_en.html

[85] http://directreadout.gsfc.nasa.gov

[86] http://www.ece.lsu.edu/vaidy/raw04/

[87] http://fpl.org

[88] http://www.iti.uni-luebeck.de/PARS/

[89] http://www.speedup.ch/

[90] G. Lienhart: Beschleunigung Hydrodynamischer N-Körper-Simulationen mit Rekonfigurierbaren Rechensystemen; Joint 33rd Speedup and 19th PARS Workshop; Basel, Switzerland, March 19 - 21, 2003

[91] Reiner Hartenstein (invited lecture): Reconfigurable Computing and its Compilation Techniques; Summer School on "Multiprocessor Systems on Chip"; Örebro, Sweden, August 25-27, 2003; Swedish National Program on Socware / Linköping University / Lund University

[92] Reiner Hartenstein (invited lecture): Distributed Memory and Datastream-based Reconfigurable Computing; Summer School on "Multiprocessor Systems on Chip"; Örebro, Sweden, August 25-27, 2003; Swedish National Program on Socware / Linköping University / Lund University

[93] Reiner Hartenstein (invited lecture): Reconfigurable Computing and its Impact on SoC and beyond; REASON Summer School on FPGA-based and Reconfigurable Systems, University of Ljubljana, Ljubljana, Slovenia, August 11-15 August 2003

[94] J. Rabaey (keynote): Silicon platforms for the next generation wireless systems; Proc. FPL 2000, Villach, Austria

[95] Reiner Hartenstein (keynote address): Software or Configware? About the Digital Divide of Parallel Computing; 18th International Parallel and Distributed Processing Symposium (IPDPS), April 26– 30, 2004, Santa Fe, New Mexico

[96] Reiner Hartenstein (keynote address): The impact of Morphware on Parallel Processing; 12-th Euromicro Conference on Parallel, Distributed and Network based Processing (PDP04); February, 11-13, 2004. A Coruña, Spain,

[97] Reiner Hartenstein (invited presentation): Morphware: neue Perspektiven für eingebettete Systeme; SFB Selbstoptimierende Systeme des Maschinenbau: Workshop Selbstoptimierung und Adaption, Paderborn, Germany, 24. - 25. November 2003

[98] Reiner Hartenstein (invited paper): Data-Stream-Based Computing: Models and Architectural Resources; International Conference on Microelectronics, Devices and Materials (MIDEM 2003), Ptuj, Slovenia, Oct.1-3, 2003

[99] Reiner Hartenstein (invited presentation): Toward Reconfigurable Computing via Concussive Paradigm Shifts; Anniversary Colloquium at Prof. Glesner's 60th Birthday; August 29, 2003, Darmstadt, Germany.

[100] Reiner Hartenstein (invited talk): Reconfigurable Computing a la Mead and Conway; Kolloquium, Informatik X, TU München; August 21 2003, Munich, Germany.

[101] Reiner Hartenstein (opening keynote): A Mead-&-Conway-like Break-through is overdue; Dagstuhl Seminar Nº 03301, Dynamically Reconfigurable Architectures; Dagstuhl, Germany, 20. 07.-25. 07. 2003,

[102] Reiner Hartenstein (invited presentation): Reconfigurable Computing and its Impact; intel ORCC, intel On Chip Reconfigurable Computing and Communication Workshop (intel ORCC workshop), Hillsboro, Oregon, USA, May 15-16, 2003

[103] Reiner Hartenstein (keynote address): Disruptive Trends by Custom Compute Engines; The 12th International Conference on Field Programmable Logic and Application FPL 2002, September 2 - 4, 2002, La Grande-Motte (Montpellier, France)

[104] Reiner Hartenstein (keynote address): Reconfigurable Computing: urging a revision of basic CS curricula; The 15th International Conf. on Systems Engineering - ICSENG02, Las Vegas, USA, 6-8 August, 2002

[105] Reiner Hartenstein (keynote address): Stream-based Computing - Antimatter of Informatics; First International Conf. on Intelligent Computing and Information Systems (ICICIS 2002), Cairo, Egypt, June 24-26, 2002.

[106] Reiner Hartenstein (keynote address): Configware / Software Co-Design: Be Prepared For the Next Revolution! The 5th IEEE Workshop on Design & Diagnosis of Electronic Circuits & Systems (DDECS'02), Brno, Czech Republic, April 17 - 19, 2002 -

[107] Reiner Hartenstein (invited presentation): Reconfigurable Computing Architectures and Methodologies for System-on-Chip; 3rd Workshop on Enabling Technologies for System-on-Chip Development 2001, November 19-20, 2001, Tampere, Finland.

[108] Reiner Hartenstein (keynote address): Reconfigurable Computing: a New Business Model - and its Impact on SoC Design; DSD'2001 - EUROMICRO Symposium on Digital System Design: Architectures, Methods, and Tools, Warsaw, Poland, September 4 - 6, 2001.