Co-Design and High Performance Computing: Scenes and Crisis

Reiner W. Hartenstein, Jürgen Becker, Michael Herz, Ulrich Nageldinger

University of Kaiserslautern
Erwin-Schrödinger-Straße, D-67663 Kaiserslautern, Germany
Fax: ++49 631 205 2640, email: abakus@informatik.uni-kl.de
http://xputers.informatik.uni-kl.de

ABSTRACT

During the development of scientific disciplines, mainstream periods alternate with revolution periods, where "out of the way disciplines" can become a mainstream. Just in the moment increasing turbulences announce a new revolution. The variety of "High Performance Computing" scenes will be mixed up. Can an increasing application of structurally programmable hardware platforms (Computing by the Yard) break the monopoly of the von Neumann mainstream paradigm (Computing in Time) also in multipurpose hardware? From a Co-Design point of view, the paper tries to provide an overview through the turbulences and tendencies, and introduces a fundamentally new machine paradigm, which uses a field-programmable data path array (FPDPA) providing instruction level parallelism. The paper drafts a structured design space for all kinds of parallel algorithm implementations and platforms: procedural programming vs. structural programming, concurrent vs. parallel, hardwired vs. reconfigurable. A structured view by rearranging the variety of computing science scenes seems to be feasible.

Key Words: Xputer, accelerator, application-specific processor, embedded system, field-programmable devices, hardware/software co-design, instruction level parallelism, non von Neumann, partitioning, rapid prototyping

1. PREFACE

The scene of "High Performance Computing" is like a band wagon. The scene gets more and more intransparent. Some players even discuss a splitting-up of the area into "Computer Science" and "Computing Science". Why do Real Time Systems, Multimedia Hardware, Digital Signal Processing, Custom Computing Machines, ASIPs, Hardware/Software Co-Design and others form isolated subscenes? Hundreds of conferences per year are held worldwide. All have the same goal: high performance, for an adequate price. The wide variety of applications does not sufficiently explain the Babylonic confusion. There is a severe lack of survey papers and good tutorials. Conflicting terminology adds to the confusion. A new IEEE Technical Committee "Engineering of Computer-Based Systems (ECBS)" tries to structure this chaos. General models are missing, which the TC-ECBS is still looking for.

Figure 1. History of turbulences in Computing, Parallel and/or High Performance Computing.
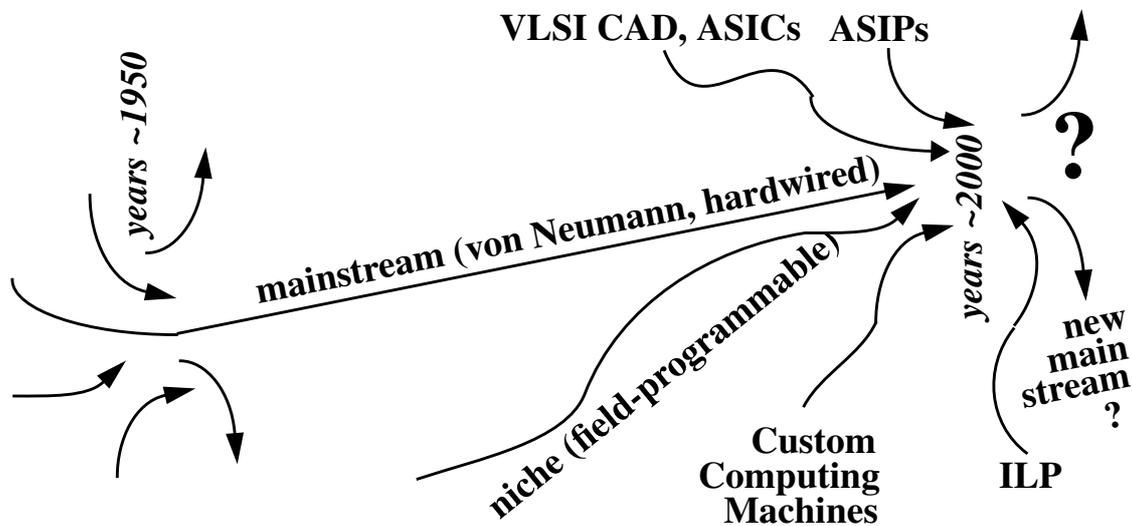

Sequential programming has an excellent paradigm, namely von Neumann. But parallel computing has been following this wrong paradigm for decades. And Hardware/Software Co-design had no paradigm at all. Panelists frankly admit, that the entire discipline is in a severe crisis. Waiting for the Terabyte-(per second-)Bus does not seem to be a way out. At the High Performance Computing-Symposium in New Delhi, in the end of 1995, field programmable logic has been officially introduced – by an opening-keynote speaker from MIT – as "Computing by the Yard" (vs. Computing in Time). This is a milestone, since shortly before high performance research has not looked at this new accelerator technology platform. From a Co-Design point of view, the paper tries to provide an overview through the turbulences and tendencies. It drafts a structured design space for all kinds of parallel algorithm implementations and platforms: procedural programming vs. structural programming, concurrent vs. parallel, hardwired vs. reconfigurable. A structured view by rearranging the variety of computing science scenes seems to be feasible.

2. INTRODUCTION

The complete *system on a chip* (SOC) is on the advance. Viewing VLSI-Design tools, the focus of public interest moves up one level of abstraction every 6 years. After polygons, transistors and the gate level (logic synthesis), the register transfer level used by high level synthesis and VHDL (mainly in Europe) or Verilog (mainly USA) has passed its peak now. Presently, the interest turns more and more to the processor level, which may reach its peak by the year 2000. All over the world, special processors and microprocessors have already achieved a higher turnover in the microchip market than universal processors.

Figure 2.   Extending the ALU through the backdoor (at CCMs).
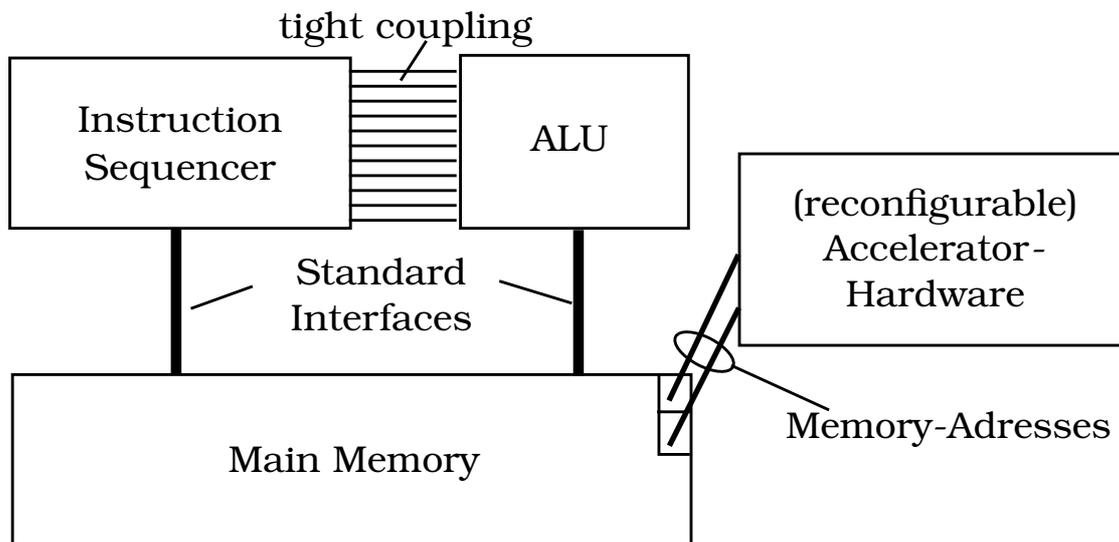
In several R&D-scenes, application-specific microprocessors are on the advance [2], [3]: as ASIPs (application-specific instruction set processors [4] - [7]), as FCCMs (FPGA-based custom computing machines [8] - [15]), or by applying of hardware-software-co-design [16] - [26]. The more radical ASIP-method creates complete application-specific instruction sets, the reason why compilers, operating-systems, simulators and emulators must be generated for every ASIP seperately. In contrast, the approaches of FCCMs and hardware-software-co-design add only a few application-specific extensions to the (standard-) instruction sets of universal processors by external accelerator hardware (AH). The advantage of these approaches is the usability of commercially available software. The disadvantage of the apparently more flexible methods of FCCMs or hardware-software-co-design appears, that the interface between AH on the one side and the host inclusive system software on the other side is only a tinker toy solution. As soon as the instruction set is changing, a new instruction sequencer is necessary because of its tight coupling to the ALU (figure 2). To avoid this, the AH must be connected for example using reserved memory addresses (figure 2).

3. REVOLUTION AGAINST VON NEUMANN?

Besides the mainstream of the sequential programmed von-Neumann-processors, the field-programmable logic (FPL) has been developed unnoticed for a long time. For example, the FPLA has been available for three decades. But meanwhile, FPL components like FPGAs have reached several billion dollars of turnover all over the world. Hardware has become soft and besides sequential programming the structural programming is getting more and more important [27] - [30]. In addition to reconfigurable components, more and more reconfigurable commercial boards appear now (e. g. [31] - [35]). An important application are embedded reconfigurable accelerators (e. g. [36] - [39]). Will reconfigurable architectures (like e. g. also [40] - [43]), i. e. structurally programmable processors, sometime overcome the von-Neumann paradigm? Or will these two worlds merge together forming a new system of theories to produce a new super paradigm? But there is still a long way to go to reach this goal. At the time, the two worlds of computing in time and computing in space are not yet married. The systolic arrays ([44], [45]), the product of their first brief rendezvous, was actually a virgin birth.
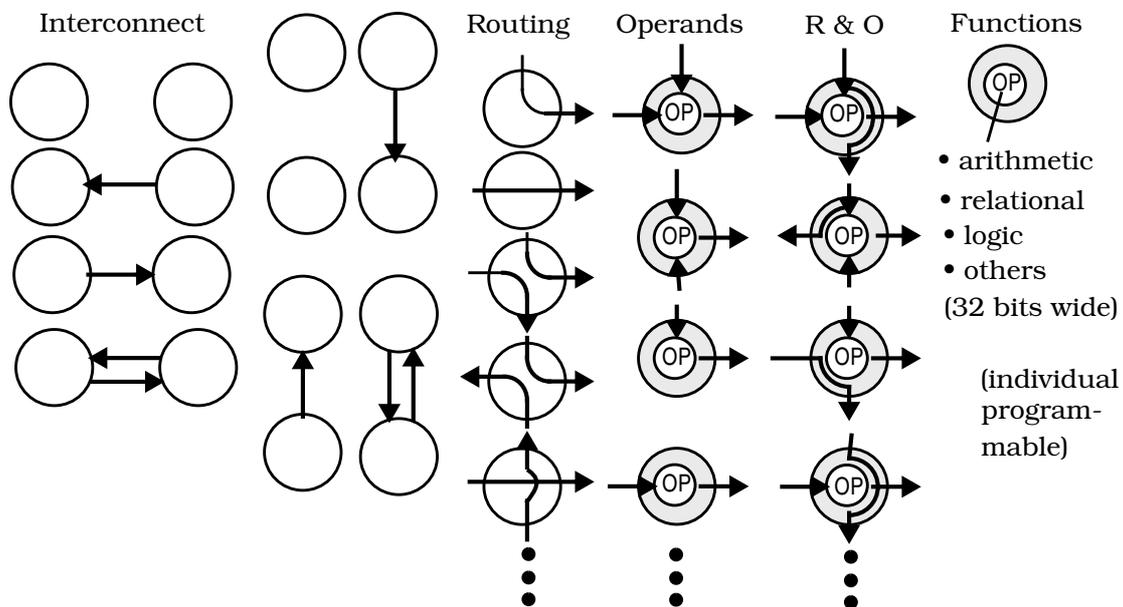
4. RECONFIGURABLE PARALLEL ALUs

Apropos arrays:

Figure 3.   Example of an rDPA (reconfigurable Data Path Array)
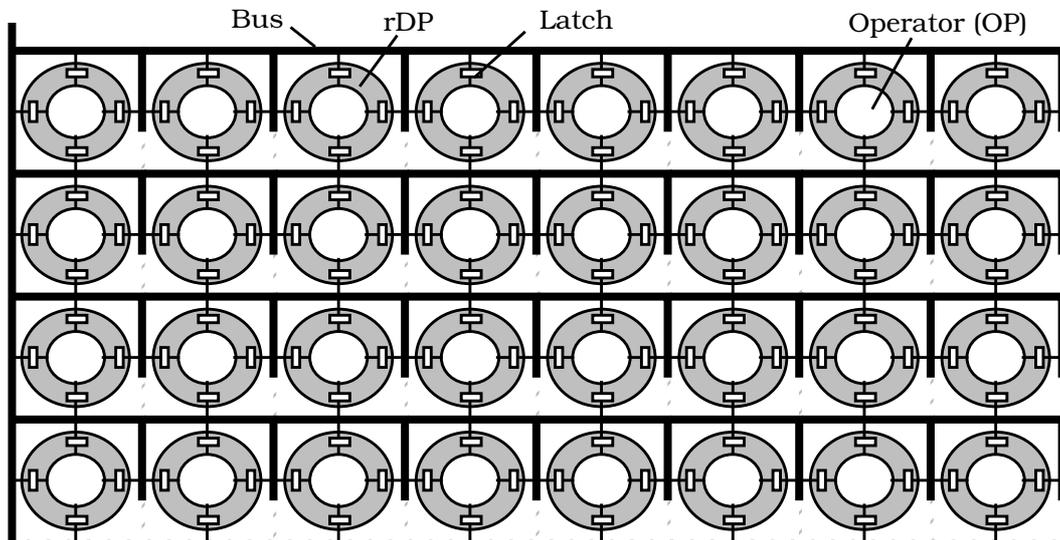
the reconfigurable arrays[46] demonstrate, that structural programming is not necessarily a domain of Sauerkraut structures, like for fast hardware prototyping.

Figure 4.   Possibilities of Programming the rDPA of figure 3.


& other mirror & rotate variants

The easy way of programming such an array will be illustrated by the rDPA (reconfigurable Data Path Array) example by Rainer Kress [47], [48]. Figure 3 shows such an array of rDPs (reconfigurable Data Paths) and figure 4 illustrates its possibilities of structural programming: the interconnect between the rDPs, the routing through one rDP, the routing additionally to the operator (OP) inside of the rDP, as well as the function of the OP. As an example of an array-processing application, the eight equations in figure 5 are used. A first attempt may result in the solution in figure 6, where 17 bus cycles are necessary, making the only available bus the

Figure 5.   An application example for the rDPA- structural programming.

$$y10 := a0 * (b0 + 2 * c0); \qquad y11 := a1 * (y10 + 2 * c1);$$
$$y20 := 5 * d0 + e0 + (f0 + b0): \qquad y21 := 5 * y20 + e1 + (f1 + y10):$$
$$y30 := g0 * (h0 + 2 * e0); \qquad y31 := y30 * (y40 + 2 * e1);$$
$$y40 := (5 * d0 + e0) * f0; \qquad y41 := (5 * y20 + e1) * f1;$$

performance bottleneck. Out of this, an optimizing algorithm, based on the method of simulated annealing, generates after a few seconds computing time the solution in figure 7, where the bus is used only for one transfer cycle. All other connections are local between directly neighboured rDPs. The possibility to program rDPs exclusively as routing-elements, i. e. without using the Operator OP inside the rDP, results in more flexibility in favour of better optimised solutions. An amazing effect of this method is the generation of good solutions without the necessity to understand them in detail. Also, the labour to design sophisticated, but possibly not enough efficient heuristics for a compiler is obsolete. In Kaiserslautern,

Figure 6.   First approach for rDPA-Routing-und-Placement for the problem in figure 3.
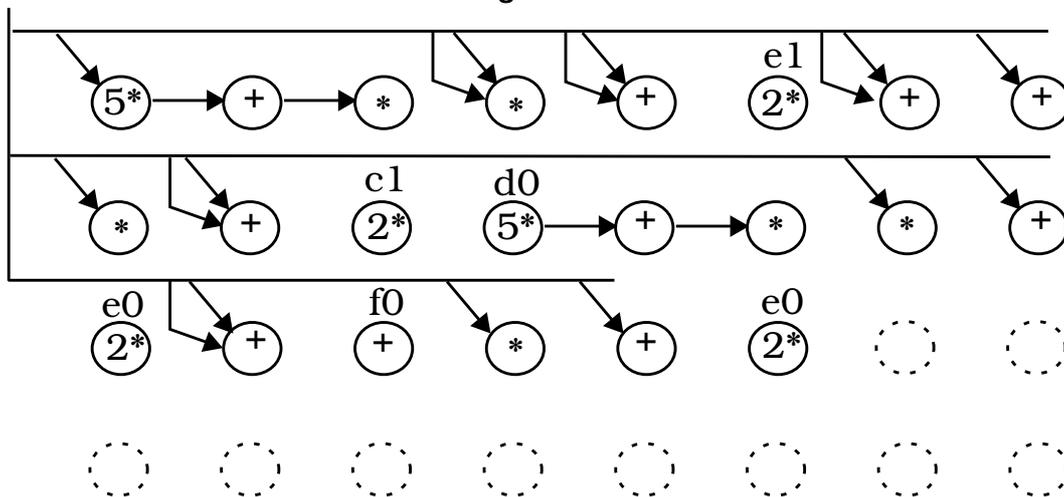
a two level partitioning compiler has been developed, which accepts a C-dialect (extension of ANSI-C) and generates both: sequential code for (von-Neumann-) host and Xputer data sequencer, as well as structural code for (re-)configuring an rDPA [63] - [65]

Figure 7.   Optimized rDPA-Routing- and-Placement- solution of the equations in figure 3.
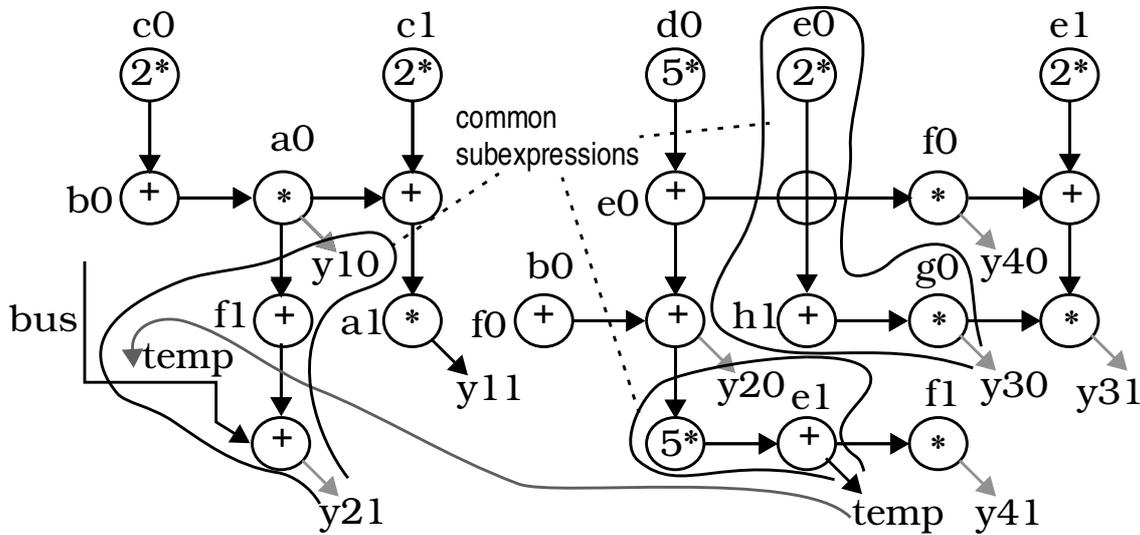

5. SEARCHING FOR NEW PARADIGMS

Established well-known disciplines normally have a structured scheme, which is a kind of 'coordinate system' for navigation in their design universe giving a rough orientation. The dimensions to this 'coordinate system' are possibly added by a general model for the basic structures of objects from these disciplines. For the classical discipline of 'Sequential Computing' a very well suited basic model is available with the von Neumann paradigm, which gives orientation for all related lecturing requirements. But how is the situation in newer disciplines of computer science and electrical engineering? What paradigms give orientation to Hardware/Software Co-Design, Parallel Computing, High Performance Computing or whatever the names of these non-classical disciplines are?

The crisis of parallel computing is obvious on many panel discussions of their conferences. The more the market is shrinking, the more drastical is the reaction in the turbulences of the inflating international academic research scene. The number of conferences of this scene is steadily growing and hardly estimatable. After loosing the touch with the goal, the efforts are mutliplied. The scene is still oriented primarily on the von Neumann paradigm. But maybe this is the reason for this crisis. The von Neumann model is a computing but no communication paradigm. Thus its level of abstraction is not high enough, so that it cannot help to navigate through so many different architectures. The von Neumann paradigm is located one abstraction level too low. Is it the fault of this wrong paradigm, that there are hardly any survey papers and that the plenty of conferences shows only diffuse goals as well as their programs reflect a chaotic grocery store? Or is the reason of the confusion only the inflation of the conferences combined with short travel etats, which leads to a strategy of identifying every non-rejected submission with a paying attendee?

But not only parallel computing tries to find high performance solutions, also the scenes of ASAPs (Application-specific Array Processors), and Systolic Arrays respectively, of Custom Computing Machines (CCMs) and Hardware/Software Co-Design deal with high performance computing. The scientific area of ASAPs is well organized and reflects a mainstream without containing the von Neumann paradigm (as only the clock generator is left over from the sequencer). Maybe this results in such a good systematic of this classical looking theory system.

Figure 8. Procedurally data-driven machine.
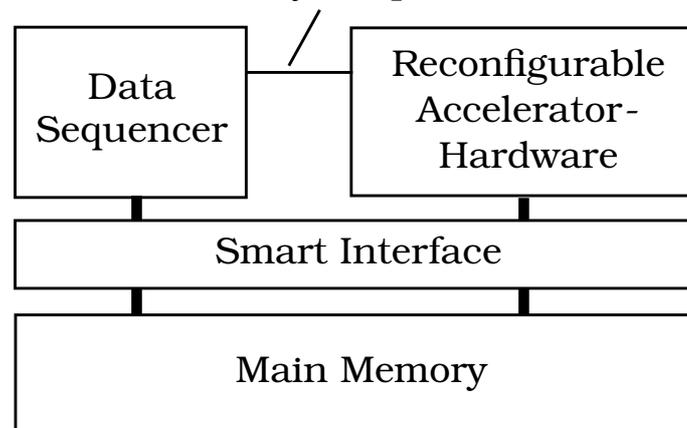
H/S Co-Design achieves a higher throughput by software-to-hardware migration, where the objects of migration are usually very frequently computed loops from the software part, which are therefore performance bottlenecks. E.g., these loops can be implemented into powerful hardware operators used as an accelerator. The main goal in H/S Co-Design (e.g. designing embedded systems) is an acceptable H/S Trade-off: achieving a required throughput as cheap (less hardware!) as possible. But H/S Co-Design is still missing a general model [17]. Thus this scene is splitted by a large variety of different architectures. The scene of FPGA-based Custom Computing Machines (F-CCMs) is close related to H/S Co-Design, because here accelerator hardware is also connected to a von Neumann host. This accelerator hardware is implementing a few powerful application-specific extensions to the instruction set of the host's universal processor. But F-CCMs concentrate more on the reconfigurable hardware platform of the accelerator, whereas H/S Co-Design is focusing on automatic partitioning methodologies. Both scenes are missing a general paradigm.

6. A NEW COMPUTING-PARADIGM BEYOND VON NEUMANN

The von Neumann paradigm is not very well suited for these two disciplines, because of its tight coupling between ALU and instruction sequencer. As soon as the instruction set of the ALU is changed, a new instruction sequencer is needed. The connection of reconfigurable accelerator hardware to the host is equivalent to changing the ALU. Thus those accelerators are connected through the 'backdoor': the accelerator is activated by addressing reserved locations of the main memory (figure 2). So the call of an accelerator function is performed in the mode 'transport-triggered' [49] - [54] instead per 'Instruction fetch'. Such an improvised interface is a tinker toy approach rather than a general systematic model.

The structural re-programming of a reconfigurable accelerator is each time a different addition to the instruction set of the ALU, which corresponds to a new ALU. But how can we avoid that partly reconfigurations result in a falling apart of the procesor architecture? Is it maybe possible to sacrifice the instruction sequencer? But without sequencing a processor cannot exist. A way out is using a data sequencer [55], [56] instead, which is only loosely coupled with the ALU via a few decision bits (figure 8). In this case activations of powerful accelerating operators are always performed in the mode 'transport-triggered', because there exists no longer a classical 'instruction fetch'. For integrating accelerators into host environments this data-driven model is useful, because here applications need the same compound operator frequently. The 'instruction fetch' is migrated now from run-time to the compile-time of the accelerator.

By exchanging instruction sequencer to data sequencer a non von Neumann processor is achieved, which is not control-flow-driven but data-flow-driven [59]. Such a data procedural processor (d-processor, named Xputer[1]) is working fully deterministically, which doesn't correspond to a so called data flow machine [57]. The execution order of such data flow machines isn't predictable, because an arbiter is determining the next executable instruction (figure 9).

Figure 9.   Comparisons of different computing paradigms

Computer (von-Neumann-Machine)	Xputer (d-Paradigm)	Dataflow-Machine
procedural Sequencing (deterministic)		Arbiter-Sequencing (indeterministic)
Control-flow-driven	Data-driven	

Instead of 'instruction-jumps' or 'statement-loops', a d-processor is performing 'data-jumps' and 'data-loops'. Thus the destinations of jumps are not instructions but data objects [58]. As well as the von Neumann paradigm guaranties universality of its processors, an existing d-paradigm garanties the universality of d-processors [59] - [62].

A d-processor is also able to work independently in a stand-alone mode [39]. But from a commercial point of view, a general substitution of von Neumann processors is not possible, because available system- and application-software is not compatible to a d-processor. But the use as co-processor to state-of-the-art workstations being a flexible or universal accelerator is possible [63] - [65].

Several different mechanisms, which will be sketched only briefly here, open an enormous acceleration potential with reasonable hardware costs for the d-processor applied to algorithms from signal- and image-processing, mutlimedia and other related fields [37] - [39]. One of these mechanisms is the elimination of different kinds of overheads as well as run-time/compile-time migration, e.g. switching many communication paths at loading time instead run-time. The importance of this migration will be emphasized by the normally unsolved explosion of communication switches in the classical massively parallel computer systems. Frequently measured speed-up factors (in the extreme case more than 3 orders of magnitude) show, that d-processors are a promising universal platform [12], [38], [55], [60] - [68].

Another mechanism is detecting and implementing parallelism on data path level into the reconfigurable devices [47], [48], [64]. The reconfigurable devices in our approach demonstrate, that structural programming represent a systematic way in synthesizing specifications efficiently into an array of reconfigurable data path units (DPUs). This was explained by the reconfigurable data path array (rDPA) approach in section 4.

The application development and support environment CoDe-X (Co-Design for Xputers) is integrating automatically d-processors into von Neumann hosts transparently for the user (see figure 10) [63], [65].

---

1. non von Neumann, not to be mixed up with the Transputer, which is a von Neumann processor

Figure 10. Overview on application development environment CoDe-X
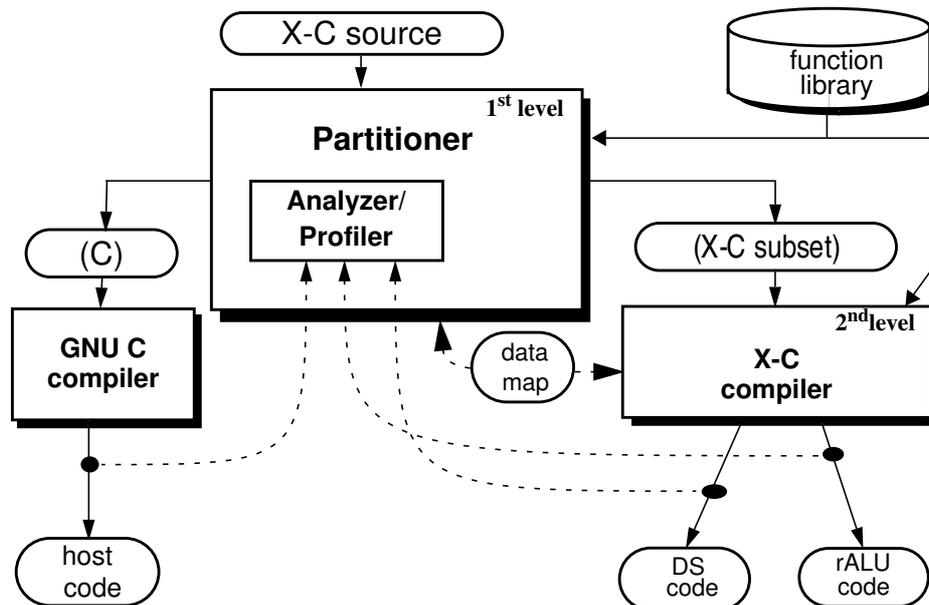
Input to the CoDe-X framework is a C-dialect, which is an extension to ANSI C including an optional data-procedural extension [58] (see function library in figure 10) for achieving highest possible acceleration factors from d-processors. So, commercially available software can be automatically accelerated by using CoDe-X and d-processors by reasonable hardware costs. CoDe-X is therefore performing a two-level partitioning approach as illustrated in figure 10:

- the first level is performing a profiling-driven Host/Xputer partitioning, where performance bottlenecks are allocated to the accelerator hardware.

- The X-C compiler in the second level is generating sequential (DS code) and structural (rALU code) by optimizing parameter-driven the Xputer hardware utilization.

7. CONCLUSIONS

Hardware has become soft and introduces the parallelism of the space-dimension, the Computing by the Yard [1] into the attention of the – until now – only von Neumann oriented, therefore mainly procedurally thinking High-Performance- und Parallel-Computing-scenes. This is a milestone and shows a possible way to the dichotomic Computing Science (computing over the time <u>and</u> in the space). The Systolic-Array-scene developed mappings, which connect both worlds. But the nearly monopolistic dominant von-Neumann-Paradigm is a Paradigm of the computing over the time, but not of the computing in the space. We have introduced a paradigm, which shows a way towards a systematic dichotomy of Computing Sciences. This novel d-paradigm integrates programming and resources for computing in both worlds and has mainstream potential. It has the chance to break the monopoly of the von Neumann paradigm as a conceptual backbone of computer science.

The conference-inflation of the High-Performance-Computing is already giving a signal for a mass movement of a general new time of departures — away from the von-Neumann-Paradigm? A new religion or only a new sect? But the direction is not known yet and there are growing doubts about the sense of clinging desperately to the von-Neumann-Paradigm supposed to be the only glorifying solution.

REFERENCES

[1]   A. Agarwal: Hot Machines; Proc. Int'l Conf on High Performance Computing; Dec. 27-30, 1995, New Delhi, India

[2]   A. Postula, D. Abramson, P. Logothethis: Synthesis for Prototyping of Application-specific Processors; Proc. 3rd Asia Pacific Conf. on Hardware Description Languages (APCHDL'96), Bangalore. India, Jan. 1996

[3]   R. Hartenstein, J. Becker, R. Kress: Application Specific Design Methodologies: General Model vs. Tinker Toy Approach; GI/ITG Workshop on Custom Computing, Schloß Dagstuhl, Germany, June 1996

[4]   A. Postula, D. Abramson, P. Logothethis: Synthesis for Prototyping of Application Specific Processors; Proc. 3rd Asia Pacific Conference on Hardware Description Languages (APCHDL´96), Bangalore, India, Jan. 1996

[5]   Ing-Jer Huang, A. Despain: Synthesis of Application Specific Instruction Sets; IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 14, No. 6, June 1995

[6]   H. Akaboshi, H. Yasuura: COACH: A Computer Aided Design Tool for Computer Architects; IEICE Trans. Fundamentals, Vol. E76-A, No. 10, Oct. 1993

[7]   J. Sato, M. Imai, T. Hakata, A. Y. Alomary, N. Hikichi: An Integrated Design Environment for Application Specific Integrated Processor; Proc. IEEE Int'l Conf. on Computer Design: ICCD 1991, pp. 414-417, Oct. 1991

[8]   D. Buell, K. Pocek: IEEE Workshop on FPGAs for Custom Computing Machines (FCCM'93) Napa, CA, April 1993

[9]   D. Buell, K. Pocek: IEEE Workshop on FPGAs for Custom Computing Machines (FCCM'94), Napa, CA, April 1994

[10]   P. Athanas, K. Pocek: IEEE Workshop on FPGAs for Custom Computing Machines (FCCM'95), Napa, CA, April 1995

[11]   J. Arnold, K. Pocek: IEEE Workshop on FPGAs for Custom Computing Machines (FCCM'96), Napa, CA, April 1996

[12]   R. Hartenstein et al.: Custom Computing Machines (invited opening keynote); DMM'95 - Int'l Symp. on Design Methodologies in Microelectronics, Smolenice Castle, Slovakia, September 1995

[13]   R. Hartenstein: Custom Computing Machines; aktuelles Schlagwort; GI Informatik-Spektrum 18: p. 228-229, Springer-Verlag, Juni, 1995)

[14]   N. N.: Brigham Young University Reconfigurable Logic Lab Bibliography; e-mail: wirthlin@fpga.ee.byu.edu, 1995

[15]   S. A. Guccione: List of FPGA-based Computing Machines; guccione@ccwf.cc.utexas.edu, last updated: June 2, 1995

[16]   K. Buchenrieder: Hardware/Software Co-Design in der Industrie; IT Press (planned for 1996)

[17]   R. Gupta: Hardware/Software Co-Design; Proc. 9th Int'l Conference on VLSI Design, Bangalore. India, Jan. 3-6, 1996

[18]   R. Gupta: Co-Synthesis of Hardware and Software for Digital Embedded Systems; Kluwer, 1995

[19]   R. Hartenstein: Hardware/Software Co-Design; aktuelles Schlagwort; GI Informatik-Spektrum 18: p. 286-287, Springer-Verlag, Oktober, 1995)

[20]   Proc. 1st Int'l Workshop on Hardware/Software Co-Design CODES/CASHE '92, Estes Park, Colorado, USA, 1992

[21]   Proc. 2nd Int'l Workshop on Hardware/Software Co-Design CODES/CASHE '93, Innsbruck, Austria, 1993

[22]   Proc. 3rd Int'l Workshop on Hardware/Software Co-Design CODES/CASHE '94, Grenoble, France, 1994

[23]   Proc. 4th Int'l Workshop on Hardware/Software Co-Design CODES/CASHE '96, Pittsburgh, USA, March 1996B. K. Fawcett: FPGAs as Configurable Computing Elements; Int'l Worksh. on Reconfigurable Architectures, @ ISPS'95 - 9th Int'l Parallel Processing Symposium, Santa Barbara, 24. - 29. April 1995

[24]   R. Gupta: Cosynthesis of Hardware and Software for Digital Embedded Systems; Kluwer 1995

[25]   R. Gupta, G. de Micheli: Hardware/Software Co-Synthesis for Digital Systems; IEEE D&T of Computers, Sept. 1993

[26]   K. Buchenrieder: Hardware/Software Co-Design; IT Press, Chicago 1995

[27]   H. Grünbacher, R. Hartenstein (Editors.): Field-Programmable Gate Arrays: Architectures and Tools for Rapid Prototyping; Second International Workshop on Field-Programmable Logic and Applications, Vienna, Austria, Aug./Sept. 1992; Lecture Notes in Computer Science 705, Springer-Verlag, 1992

[28]   R. Hartenstein, M. Servít (Editors.): Field-Programmable Logic: Architectures, Synthesis and Applications; Fourth International Workshop on Field-Programmable Logic and Applications, Prague, Czech Republic, Sept. 1994; Lecture Notes in Computer Science 849, Springer-Verlag, 1994

[29]   W. Moore, W. Luk (Editors.): Field-Programmable Logic and Applications; Fifth International Workshop, Oxford, United Kingdom, Aug./Sept. 1995; Lecture Notes in Computer Science 975, Springer-Verlag, 1995

[30]   M. Glesner, R. Hartenstein (Editors.): Field-Programmable Logic and Applications; Sixth International Workshop, Darmstadt, Germany, Sept. 1996; Lecture Notes in Computer Science, Springer-Verlag, 1996

[31]   N. N.: WILDFIRE Custom Configurable Computer WAC4010/16; Document # 11502-0000, Rev. C, Annapolis Micro Systems, Inc., April 1995

[32]   P. Chan: A Field-Programmable Prototyping Board: XC4000 BORG User's Guide; UCSRC-CRL-94-18, April 1994

[33] S. Casselman, J. Schewel, M. Thornburg: H.O.T. (Hardware Object Technology) Programming Tutorial; Release 1, Virtual Computer Corporation, January 1995

[34] H. Chow, S. Casselman, H. Alunuweiri: Implementation of a Parallel VLSI Linear Convolution Architecture Using the EVC1; in: [10]

[35] N.N.: EVC-1 Info 1.1; Virtual Computer Corporation, 1994

[36] A. Koch, U. Golze (Technical University Braunschweig, Germany): A Universal Co-Processor for Workstations; in: W. R. Moore, W. Luk (eds.): More FPGAs; Abbington EE&CS Books, Oxford, UK 1993 (selection from Proc. Int'l Symposium on field-programmable Logic and Applications, Oxford, UK, Sept. 1993)

[37] J. Becker, R. Hartenstein, R. Kress, H. Reinig: High-Performance Computing Using a Reconfigurable Accelerator; Proc. Workshop on High Performance Computing, Montreal, Canada, July 1995

[38] R. Hartenstein, J. Becker, R. Kress, H. Reinig: High-Performance Computing Using a Reconfigurable Accelerator; CPE Journal, Special Issue of Concurrency: Practice and Experience, John Wiley & Sons Ltd., 1996 (invited reprint of [37])

[39] R. Hartenstein, J. Becker, R. Kress: An Embedded Accelerator for Real Time Image Processing; 8th EUROMICRO Workshop on Real Time Systems, L'Aquila, Italy, June 1996

[40] S. Guccione, M. Gonzales: Classification and Performance of Reconfigurable Architectures; FPL´95 - Int'l Symposium on field-programmable Logic and Applications, Oxford, UK, 29-31 August 1995

[41] B. K. Fawcett: FPGAs as Configurable Computing Elements; Int'l Workshop on Reconfigurable Architectures @ IPPS´95 - 9th Int'l Parallel Processing Symposium, Santa Barbara, CA, 24-29 April 1995

[42] A. Koch, U. Golze (Technical University Braunschweig, Germany): A Universal Co-Processor for Workstations; in: W. R. Moore, W. Luk (eds.): More FPGAs; Abbington EE&CS Books, Oxford, UK 1993 (selection from Proc. Int'l Symposium on field-programmable Logic and Applications, Oxford, UK, Sept. 1993)

[43] R. Gupta, G. De Micheli: Hardware-Software Cosynthesis for Digital Systems; IEEE Design & Test, Sept. 1993

[44] S. Y. Kung: VLSI Array Processors; Prentice-Hall, 1988

[45] N. Petkov: Systolische Algorithmen und Arrays; Akademie-Verlag, Berlin 1989

[46] P. Treleaven, M. Pacheco, M. Vellasco: VLSI Architectures for Neural Networks, IEEE Micro, Vol. 9, Nr. 6

[47] R. Hartenstein, R. Kress: A Datapath Synthesis System for the Reconfigurable Datapath Architecture; Asia and South Pacific Design Aut. Conf., ASP-DAC'95, Nippon Convention Center, Makuhari, Chiba, Japan, Aug. 29 - Sept. 1, 1995

[48] R. Kress: A Fast Reconfigurable ALU for Xputers; Ph.D. Thesis, University of Kaiserslautern, 1996

[49] G. J. Lipovski: On a Stack Organization for Microcomputers; in [50]

[50] R. Hartenstein, R. Zaks: Microarchitecture of Computer Systems; North Holland 1975

[51] H. Corporaal, H. Mulder: MOVE: A framework for high-performance processor design; Proc. Supercomputing ´91, Albuquerque, IEEE Computer Society Press, November 1991

[52] H. Corporaal, P, van der Arend: MOVE32INT, a Sea of Gates realization of a high performance Transport Triggered Architecture; Microprocessing and Microprogramming vol. 38, pp. 53-60, North-Holland, 1993

[53] H. Corporaal: Evaluating Transport Triggered Architectures for scalar applications; Transport Triggered Architecture; Microprocessing and Microprogramming vol. 38, pp. 45-52, North-Holland, 1993

[54] H. Corporaal: Transport Triggered Architectures; Ph. D. thesis, Tech. University of Delft, Holland, 1995

[55] R. Hartenstein, A. Hirschbiel, M. Weber: MoM - a partly custom-designed architecture compared to standard hardware; Proc. COMP EURO, Hamburg, Germany, 1989; IEEE Press 1989

[56] A. Hirschbiel: A Novel Processor Architecture Based on Auto Data Sequencing and Low Level Parallelism; Ph.D. Thesis, University of Kaiserslautern, 1991

[57] Theo Ungerer: Datenflußrechner; Teubner, 1993

[58] A. Ast, J. Becker, R. W. Hartenstein, R. Kress, H. Reinig, K. Schmidt: Data-procedural Languages for FPL-based Machines; 4th Int'l Workshop on Field Programmable Logic and Appl.ications, FPL'94, Prague, Sept. 7-10, 1994, Lecture Notes in Computer Science, Springer, 1994

[59] R. Hartenstein, A. Hirschbiel, K. Schmidt, M. Weber: A novel Paradigm of Parallel Computation and its Use to implement Simple High-Performance Hardware; Future Generation Computing Systems 7 (1991/92), (invited reprint of [60])

[60] R. Hartenstein, A. Hirschbiel, M.Weber: A Novel Paradigm of Parallel Computation and its Use to Implement Simple High Performance Hardware; InfoJapan'90- International Conference memorating the 30th Anniversary of the Computer Society of Japan, Tokyo, Japan, 1990

[61] R. W. Hartenstein, M. Riedmüller, K. Schmidt, M. Weber: A Novel Asic Design Approach Based on a New Machine Paradigm; Special Issue of IEEE Journal of Solid State Circuits on ESSCIRC´90, July 1991

[62] R. Hartenstein, M. Riedmüller, K. Schmidt, M. Weber: A Novel ASIC Design Approach based on a New Machine Paradigm; IEEE Journal of Solid State Circuits, July 1991 (eingeladener Nachdruck von [61])

[63] R. Hartenstein, J. Becker, R. Kress: Two-Level Hardware/Software Partitioning Using CoDe-X; Int'l IEEE Symp. on Engineering of Computer Based Systems (ECBS), Friedrichshafen, Germany, March 1996

[64] K. Schmidt: A Program Partitioning, Restructuring, and Mapping Method for Xputers; Ph.D. Thesis, Kaiserslautern 1994

[65] Reiner W. Hartenstein, Jürgen Becker, Michael Herz, Rainer Kress, Ulrich Nageldinger: A Parallelizing Programming Environment for Embedded Xputer-based Accelerators; High Performance Computing Symposium '96, Ottawa, Canada, June 1996

[66] R. W. Hartenstein, R. Kress, H. Reinig: A Dynamically Reconfigurable Wavefront Array Architecture for Evaluation of Expressions; Proc. Int'l Conference on Application-Specific Array Processors, ASAP'94, San Francisco, IEEE Computer Society Press, Los Alamitos, CA, Aug. 1994

[67] Reiner W. Hartenstein, Jürgen Becker, Rainer Kress, Helmut Reinig, Karin Schmidt: A Reconfigurable Machine for Applications in Image and Video Compression; Conference on Compression Technologies and Standards for Image and Video Compression, Amsterdam, The Netherlands, March 1995

[68] Jürgen Becker, Reiner W. Hartenstein, Rainer Kress, Helmut Reinig: A Reconfigurable Parallel Architecture to Accelerate Scientific Computation; Proc. Int'l Conf. on High Performance Computing, New Delhi, India, December, 1995