# Reconfigurable Supercomputing

aktuelles Schlagwort

Reiner Hartenstein

**Die vor etwa 20 Jahren aufgekommenen FPGAs (Field-Programmable Gate Arrays) galten zunächst als alternative Plattformen für den Logik-Entwurf. Als nunmehr 6 Mrd. US-Dollars erreichendes am schnellsten wachsendes Segment des Halbleitermarktes und in eingebetteten Systemen längst „mainstream" geworden, sieht man diese Methodologie nun auch gern aus einer höheren Abstraktions-Ebene: als Reconfigurable Computing. Dies ist die Grundlage einer innovativen Form der Implementierung vieler Anwendungen nicht nur für eingebettete Systeme, sondern nun auch in allen Gebieten wissenschaftlichen Rechnens. Da neuerdings Anbieter wie Cray und silicon graphics mit FPGAs ausgerüstete Module nebst Bibliotheken anbieten, ist dies als Reconfigurable Supercomputing ein aktuelles Thema.**

## Ein Paradigmen-Wechsel

„Is it time to scrap the von Neumann architecture?" fragt Anup Gosh, vom Advanced Technology Office [1]. Jedenfalls wird die Dominanz des von Neumann (vN) Paradigma im geistigen Zentrum der Informatik bald ihren Kulminationspunkt überschritten haben [2] [4]. Die zunehmende Zahl quasi als Prothesen dienender (nicht-von-Neumann-) Akzelerator-Ko-Prozessoren demonstrieren dessen zunehmende Senilität. Dabei haben ***FPGA***s (Field-Programmable Gate Arrays) [5] wegen Ihrer Flexibilität die spezifisches Silizium erfordernden ***ASICs*** (Application-specific Integrated Circuits) weitgehend von Markt verdrängt. Die Allgegenwart von FPGAs für ***Reconfigurable Computing*** auf allen Anwendungsgebieten [6] rechtfertigt nicht mehr die Bewertung als aktuelles Schlagwort. Aber etwa seit Mitte dieses Jahres ist hieraus das Thema des ***Reconfigurable Supercomputing*** aktuell geworden [8].

Aus der Vogelperspektive erscheint diese Entwicklung als Paradigmenwechsel, was durch Nick Tredennick's Klassifikations-Schema veranschaulicht wird [9] (Bild 1). Während Architekturen nach von-Neumann-Paradigma durch Befehlsströme getrieben werden und nur aus einer einzigen Art von Quellen (Software) programmiert werden, müssen Rekonfigurierbare Plattformen nach zweierlei Arten von Quellen programmiert werden: zuerst aus einer ***Configware***-Quelle zwecks Konfiguration zur Betriebsbereitschaft, d. h. zur strukturellen Programmierung, also Programmierung im Raum, anstatt über der Zeit, und sodann aus einer ***Flowware***-Quelle zur Generierung von Flowware-Kode zur Implementierung der Aktivierung der Datenströme. Wegen mangelndem Terminologie-Konsens sei darauf hingewiesen, daß sich hier der Begriff des Datenstrom an die „***data stream***"-Definition aus der Szene der systolischen Arrays anlehnt [10] [11].

Der vom Software-Begriff sich deutlich distanzierende Terminus „Configware" macht Sinn aus semantischen Gründen, da hier grundlegend andere Übersetzer und Methodologien benötigt werden, als für Software. Im Paradigma kommt kein Programmzähler vor. Und der durch den Configware-Compiler generierte Konfigurations-Kode ist ein Strukturplan, jedoch —im Gegensatz zu Objekt-Kode aus Software— kein Befehlsabruf-Fahrplan. Sobald die Ressourcen konfiguriert sind, muß aus der Flowware-Quelle der Datenfluß-Kode generiert werden: der „Fahrplan" für die von der konfigurierten Ressource benötigten Datenströme, damit die Betriebs-Phase beginnen kann, die von Datenströmen getrieben wird, denn bei ***einfachen*** rekonfigurierbaren Plattformen gibt es zur Laufzeit keinerlei Befehlszugriff.

## Configware-Betriebssysteme

Obiger Satz (während der Laufzeit werden keine Befehle geladen) sollte je nach Background und Perspektive des Lesers vielleicht relativiert werden, da in der Praxis oft hybride Mischmasch-Systeme im Vordergrund stehen, oder ***komplexe*** rekonfigurierbare Systeme, wie z. B. ***dynamisch rekonfigurierbar***e Systeme [12], die auf Grund ihrer Segmentierung und rasch wechselnden lokalen Betriebs-Phasen immer wieder abwechselnd den einen oder den anderen Paradigmen-Hut aufsetzen. Bei dynamisch rekonfigurierbaren Systemen sind zu verschiedenen Zeitabschnitten mehrere verschiedene Configware-Kode-Makros im FPGA resident und werden je nach Bedarf während des Betriebes einzeln konfiguriert und wieder deaktiviert, bzw. überschrieben. Für die Methodologie einer ordnungsgemäßen Verwaltung, Optimierung und Steuerung dieser partiellen lokalen Eingriffe zur Laufzeit gibt es eine etablierte Forschungsrichtung, wofür der Begriff Configware-Betriebssysteme (oder „Configware/Flowware-Betriebssysteme"), als Gegensatz zu herkömmlichen Software-Betriebssystemen sinnvoll wäre, sich aber nur zögerlich durchsetzt [13]. Noch komplizierter wird die Didaktik bei Systemen, bei denen unter anderem mehrere aus Configware-Quellen konfigurierte von-Neumann-Prozessoren „soft cores" wie beispielsweise der MicroBlaze [14] [15] oder der NIOS [16] auf einen FPGA konfiguriert worden sind. Für die Praxis

wären geeignete Configware-Betriebssysteme bei derart komplexen Misch-Systemen unverzichtbar. Die von Tredennick gezeigte strenge Trennung der zwei verschiedenen Arten von Quellen und von Objekt-Kode ist bei den heute verfügbaren Programmier-Hilfsmitteln wegen fehlender oder schlecht strukturierter Kode-Modularisierung in niedrigeren Abstraktions-Ebenen nur noch äußerst schwer wieder extrahierbar. In der Logik-Ebene sehen sich alle Gatter und Flipflops doch so ähnlich.

## Das Gegenstück zum von Neumann Maschinen-Paradigma

| frühe historische Rechner | Programmierungs-Quelle |
|---|---|
| Ressourcen fest | keine |
| Algorithmen fest | keine |

| von Neumann Rechner | Programmierungs-Quelle |
|---|---|
| Ressourcen fest | keine |
| Algorithmen variabel | Software (Befehlsströme) |

| Reconfigurable Computing | Programmierungs-Quelle |
|---|---|
| Ressourcen variabel | Configware(Konfiguration) |
| Algorithmen variabel | Flowware (Datenströme) |

Bild 1: Nick Tredennick‘s Paradigmen-Klassifikation.

Für eine Diskussion der oben erwähnten Komplexitäten bietet dieser Artikel leider nicht genug Raum. Für ein allgemeines Verständnis einer breiten Leserschaft sollte stattdessen das Grundmodell in den Vordergrund gestellt werden. Für didaktische Zwecke eignet sich eine Verallgemeinerung [17] des systolischen Array [10] [11], wobei meist mehrfache Datenströme durch ein Pipe-Netzwerk hindurchströmen. Dieses Modell ist dann besonders realistisch, wenn die ***grobkörnig rekonfigurierbaren rDPAs*** (reconfigurable Data Path Arrays, wie beispielsweise XPP-III-Plattform der PACT AG in München [18]) verwendet werden, statt der feinkörnig rekonfigurierbaren FPGAs. Ein rDPA ist ein Feld von ***rDPU***s (reconfigurable Data Path Units). Eine ***DPU*** im Gegensatz zu einer CPU hat weder einen Programm-Zähler noch eine sonstige Ablaufsteuerung. Die Auslösung der Rechenoperation einer DPU erfolgt ***„transport-triggered“***, d. h. per Handshake durch Ankunft der Daten an einem der DPU-Eingänge. Durch Konfiguration wird ein rDPA zu einem Anwendungs-spezifischen Pipe-Netzwerk von rDPUs formiert. Beispielsweise bei Multimedia- oder Bildverarbeitungsanwendungen, können alle Datenströme durch on-Chip ***ASM-Speicher-Module*** generiert werden, soweit diese nicht direkt aus der Anwendung stammen. ASM steht für „auto-sequencing memory“, wobei jeder ASM-Modul einen rekonfigurierbaren Addressgenerator mit „***Datenzähler***“ hat, der schnelle Adressrechnung durchführt ***ohne Verbrauch von Speicherzyklen*** [19]. Der Paradigmenwechsel wird dadurch deutlich, daß statt des Programmzählers der (von Neumann) CPU (ein oder) mehrere Datenzähler in den ASM-Speicher-Moduln sitzen: das Reconfigurable Computing Grundmodell.

## Das Reconfigurable Computing Paradox

Im Vergleich zu Mikroprozessoren haben FPGAs viel schlechtere technologische Parameter. Sie sind Energie-Verschwender (im Vergleich zu fest verdrahteten Schaltungen) und haben erheblich niedrigere Taktfrequenzen als Mikroprozessoren. Wegen Verdrahtungs-Overhead und Rekonfigurierbarkeits-Overhead bleibt die effektiv allein für die Anwendung nutzbare Integrationsdichte (Transistoren je Chip) um mehr als 4 Größenordnungen hinter der Gordon-Moore-Kurve zurück [20]. Eine Vielzahl von Veröffentlichungen (Referenzen in [21]) berichtet hingegen über Migrationen von Software zu Configware, d. h. von Mikroprozessoren und Parallelrechnersystemen hinüber zu FPGAs, bei denen Speed-up-Faktoren um etwa eine, zwei, oder drei Größenordnungen oder sogar Faktoren bis 6000 mitgeteilt werden (Bild 2).

Diese ungeheure Diskrepanz von fast 8 Größenordnungen gilt allerdings nur für die ***feinkörnig*** rekonfigurierbaren Universal-FPGAs (Bild 3a). Feinkörnig heißt, das die als LUT (look-up table) implementierten Tausende von Konfiguierbaren Logik-Blöcken (CLBs) nur eine Pfadbreite von einem Bit haben. Für moderne, sogenannte ***Plattform-FPGA***s (Bild 3b), ist die effektive Integrationsdichte dramatisch besser, da das sehr Flächen-ineffiziente feinkörnige FPGA-Geflecht nur einen Teil der Chipfläche einnimmt. Ein Großteil der Fläche ist mit sehr Flächen-effizienten fest verdrahteten Komponenten reichhaltig bestückt wie beispielsweise Addierer, Multiplizierer, Gleitkomma-Einheiten, RAM- Speicherblöcke, Serializer-Deserializer (SerDes: sehr schnelle Ein-/Ausgabe-Schnittstellen), u. v. a. m. Die effektive Integrationsdichte von rDPAs (siehe oben) kommt der Gordon-Moore-Kurve am nächsten (Bild 3c).

Von einer erschöpfenden Erklärung dieses Phänomens sind wir aber noch weit entfernt. Dieses „***von Neumann Syndrom***“ entspringt dem jahrelangen Hängenbleiben in einer Paradigmen-Falle [22]. Mit steigender Parallelität rasch schwindende Programmierer-Produktivität durch fehlende geeignete Sprachen und Werkzeuge [23] ist nur eines von vielen ungelösten Problemen. Bei von-Neumann-basierten Hochleistungs-Rechnern und Supercomputern ist nicht die verfügbare Anzahl von ***Mikroprozessor-“Cores“*** das Hauptproblem, sondern die ***beschränkte Skalierbarkeit*** der Parallelität sequentieller Prozesse, wobei ***Amdahls Law*** nur einen kleinen Ausschnitt der Problematik erklärt. Ein im Vordergrund stehendes Hauptproblem ist auch der stark Overhead-trächtige Transport von Daten und Nachrichten [24]: der schwere

Konzertflügel wird zum Hocker des Pianisten geschoben. Aber bei rekonfigurierbaren Systemen wird der leichte Hocker zum Konzertflügel bewegt: anstelle der Daten wird der Ort der Ausführung einer Operation bewegt (vor der Rekonfigurationszeit), und zwar optimiert für eine Minimierung des Transport von Daten. Mangels Paradigmenwechsel sind aber ***rekonfigurierbare von-Neumann-Mikroprozessoren*** (Bild 3d-f, untersucht in [25]), keine Therapie des von-Neumann-Syndroms bei hochparallelen Rechensystemen.

## Mehr Rechenleistung pro Watt durch Reconfigurable Computing

Bisher war der Stromverbrauch nur ein Thema spezieller ***Low-Power-Design***-Konferenzen [26] [27] über Schaltungen in portablen Geräten, die mit einer kleinen Batterie möglichst viele Tage lang auskommen sollen. Eine sehr interessante Nebenwirkung bei Software-zu-Configware-Migration wird von der Portation einer „oil and gas“ Anwendung von einem Supercomputer auf FPGAs berichtet. Neben einen Speed-up-Faktor von deutlich mehr als einer Größenordnung wird neben einer drastischen Reduktion der Hardware-Kosten eine ***Verminderung der Stromrechnung*** auf weniger als 10% der ursprünglichen Summe angegeben [28] [29]. Leider wird bei anderen Publikationen solcher Migrationsprojekte der Einfluß auf den Stromverbrauch nicht erwähnt, denn die Diskussion über den Stromverbrauch von Computern und Supercomputern wurde erst kürzlich losgetreten [7]. Noch weitgehend unbekannt war die erst jetzt herausgesickerte Befürchtung, daß beim für die Zeit um 2010-2012 vorhergesagten ***Petaflop-Supercomputer*** ein Stromverbrauch in astronomischen Dimensionen ein Hauptproblem sein könnte. Neuerdings hat die Politik dieses Thema aufgegriffen, wie beispielsweise eine kürzliche Vorlage des Kongress der USA und eine Vorlage des US-Sentats über den Stromverbrauch von Servern, deren Markt hohe Wachstumsraten hat. Unter mehreren Prognosen über den Energie-Verbrauch der gesamten ***Cyber-Infrastruktur*** der USA (Computer und andere Elektronik überall: auch eingebettete Systeme, PCs, Unterhaltungs-Elektronik, Server-Farmen, Supercomputer etc.) geht die pessimistischste Variante davon aus, daß diese bis zum Jahre 2025 fast die Hälfte des gesamten Energieverbrauchs der USA schlucken würde. Es ist aber zu hoffen, daß für nicht portable Rechensysteme die oben genannte Reduktion der Stromrechnung um nur eine Größenordnung nur ein Anfang ist. Ermutigend sind Ergebnisse der Molecular Dynamics Machine (MDM) [30], gutes Vorbild für grobkörnig rekonfigurierbare Systeme, mit nur 0,2 Watt/GFLOPS (der Pentium 4 braucht 14 Watt/GFLOPS). .

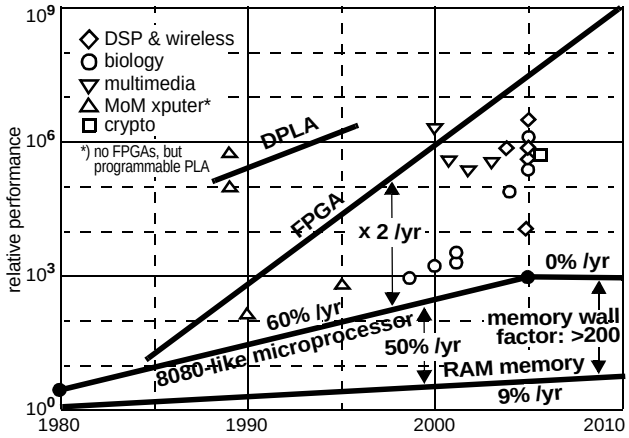

Bild 2: Speedups durch Software/Configware Migration.

## Die Krise der ESL-Industrie

Derzeit werden FPGA-basierte Implementierungen statt von Programmierern eher von ***ESL***-Entwicklern durchgeführt (ESL steht für „electronic system-level“). Die ESL-Werkzeuge-Industrie stiftet aber eine Menge Verwirrung indem sie Werkzeuge abietet, ohne zuvor eine Methodologie des Entwurfs-Verlaufes entwickelt zu haben [32]. Dabei ist die Anzahl der ESL- Entwickler um mehrere Größenordnungen höher als die der Entwickler integrierter Schaltungen, die auch von diesem Industriezweig bedient werden. Die ESL-Werkzeuge-Industrie ist hoch zersplittert und für die rasch zunehmende Komplexität der Entwürfe können die angebotenen Werkzeuge die wachsenden Anforderungen an Funktionalität, Zuverlässigkeit, Effizienz und Fehlertoleranz für Kraftfahrzeuge, Luft- und Raumfahrt, und viele andere Anwender-Industrien kaum erfüllen. Rückrufe sowie Verzögerungen in der Lancierung angekündigter Produkte demonstrieren die Notwendigkeit neuer ESL-Entwurfs-Werkzeuge allein schon zur Vermeidung kostspieliger Nachbesserungen und für eine schnellere Markteinführung neuer Produkte.

Wegen des hohen Risikos für eine Befriedigung der Mannigfaltigkeit der Landschaft der System-Industrie, kamen bisher die meisten ESL-Bemühungen aus dem akademischen Bereich (wie übrigens auch Configware-Compiler [33] [34]) und von Startups, da an sonsten Visionen darüber fehlten, was ein ESL Entwurfs-System sein sollte und wie die verschiedensten Werkzeuge in eine Gesamt-Methodologie passen sollten. Trotz der enormen Wichtigkeit kommt es erst jetzt zu einer systematischen Kategorisierung von ESL Entwurfs-Ansätzen [35], wozu natürlich auch die Einbeziehung der Anwendung von FPGAs und grobkörnig rekonfigurierbaren Plattformen gehört, die um Größenordnungen bessere Technologie-Parameter haben als FPGAs.

## Die Krise der Lehre

Ein gravierendes Problem ist die Kommunikations-Blockade zwischen der ESL-Entwurfs-Gemeinschaft und der Szene der Werkzeuge für die Programmierung hochparalleler Computersysteme. In weniger als

| # | Kategorie | Granularität | Ressourcen | effektive Integrationsdichte |
|---|---|---|---|---|
| a) | Universal-FPGA | feinkörnig | nur FPGA mit look-up tables (LUT) | sehr niedrig |
| b) | „Plattform-FPGA“ | gemischt | im FPGA: Addierer, Multiplizierer, Gleitkomma, Speicherblöcke, SerDes-Ein/Ausgabe, etc. | gut, aber Domänen-spezifisch |
| c) | rDPA | grobkönig | Pipe-Netzwerk von rDPUs | sehr gut, aber Domänen-spezifisch |
| d) | rekonfigurierbare vN Mikroprozessoren (kein Paradigmen-Wechsel*, da vN) | feinkörnig | rekonfigurierbare Befehlssatz-Erweiterung: FPGA und grobkörnige Elemente | relativ niedrig |
| e) | | gemischt | | befriedigend bis gut* Domänen-spezifisch |
| f) | | grobkörnig | | |

*) das von-Neumann-Syndrom wird nicht behoben

Bild 3: Kategorien rekonfigurierbarer Plattformen

10 Jahren werden Standardmikroprozessoren mit mehr als 128 Prozessoren auf einem Chip aufweisen [23]. Neue Programmiersprachen und -modelle sind gefordert. Herkömmliche parallele Programmierung ist bekanntlich schwierig und Fehleranfällig [23]. Die Ausbildung für künftige Informatiker muß das Thema Parallelismus starker berücksichtigen, als dies bei bisherigen Kurrikula der Fall ist [23]. Dies muß auf der Dualität der Paradigmen aufbauen: Parallelität in der Software-Domäne, kombiniert mit Parallelität durch Configware-Implementierung. Neue Programmiersprachen und Compiler sind erforderlich, welche die Vorteile beider Paradigmen miteinander verbinden: Programm-Parallelisierung mit Daten-Parallelisierung. Unsere künftigen Absolventen müssen zu S***oftware/Configware Partitionierungs-Entscheidungen*** befähigt werden [21]. Wenn diese 128 Mikroprozessoren nur als CPUs betrieben werden können, ist bei Weitem nicht mit der Effizienz der bisher publizierten Configware-Lösungen zu rechnen. Vielmehr sollte dabei jeder Prozessor wahlweise im CPU-Modus betrieben werden können, oder im rDPU-Modus: zwecks Nutzung der Vorteile grobkörnig rekonfigurierbarer Lösungen, kombiniert mit nebenläufiger Parallelität einiger CPUs auf dem gleichen Chip zur Weiterverwendung und Pflege bewährter „legacy Software“. Die Frage ist, ob Fa. intel oder AMD die Visionen hierzu aufbringen kann, oder ob wir auf von der nächsten Generation gegründete Startups warten müssen.

Aus eingebetteten Systemen nicht mehr wegzudenkende FPGAs nebst allen damit zusammenhängenden Synonyma kommen in der viele Hundert Seiten langen Version 2004 der hoch respektierten Empfehlungen der ACM / IEEE-CS / AIS ***joint task force on CS curricula*** überhaupt nicht vor [36]. In die bisher letzte Version (2005) wurde auf meine Zuschrift hin im letzten Absatz des Übersichts-Bandes ein Nebensatz hineingezwängt, sodaß „FPGA“ nun insgesamt ein einziges Mal vorkommt. Die Struktur der gesamten Empfehlungen wurde dahingehend aber überhaupt nicht geändert. Da bei Weitem die meiste Software heute für eingebettete Systeme implementiert wird, müssen diese Empfehlungen als hochgradig veraltet gesehen werden: den heutigen Arbeitsmarkt unserer Absolventen weitgehend verfehlend [37]. Bedenklich ist dabei, daß auch aus diesem Gremium Mitglieder von Akkreditierungs-Institutionen rekrutiert werden.

## Schlußfolgerungen

Rein technologisch gesehen ist der Personal Supercomputer, desktop oder laptop, heute bereits machbar, dank Reconfigurable Computing [24]. Doch bis zu einem breiten Durchbruch sind schwerwiegende soziologische Hürden zu überwinden. Kommunikations-Barrieren und Bildungs-Defizite behindern den notwendigen Paradigmenwechsel in den Köpfen. Die klassische Supercomputing-Szene wird die Entwertung ihrer immensen geistigen und finanziellen Investitionen zu verzögern (vgl. Bild 3 d-f) oder gar zu verhindern versuchen. Mitunter mußte in vergleichbaren Situationen der Durchbruch auf die nächste Generation warten [38]. Diese muß dazu aber befähigt werden: durch eine rasche Aktualisierung der Kurrikula, und zwar ab dem ersten Semester mit einem ***Doppel-Paradigmen-Ansatz***, nicht nur durch Spezialveranstaltungen, die nur von wenigen Freaks frequentiert werden.

## Literatur


[1] J. Jackson: DARPA takes aim at IT sacred cows; Government Computer News (GCN), 11. März. 2004 - http://www.gcn.com/online/vol1_no1/25240-1.html

[2] G. D. Peterson, D. Bouldin: Improving Undergraduate Computer Science and Engineering Education: A White Paper Concerning Integrative Computing Education and Research; in: O. Garcia, L. Cassel, K. Swigger (editors): Report on the NSF workshop on Integrative Computing Education and Research (ICER): Preparing IT Graduates for 2010 and Beyond; October 27-28, 2005, Dallas, Texas - URL: [3]

[3] http://www.eng.unt.edu/ICERWorkshop/executive-summary1028.pdf

[4] R. Hartenstein: Basics of Reconfigurable Computing; in: J. Henkel, S. Parameswaran (editors): Embedded Computing - A Low Power Perspective, Springer-Verlag, 2007

[5] N.N.: Field-Programmable Gate Array; http://de.wikipedia.org/wiki/FPGA

[6] N.N.: Pervasiveness of Reconfigurable Computing; http://fpl.org/click/

[7] R. Hartenstein (invited paper): Reconfigurable Supercomputing: Hurdles and Chances; International Supercomputing Conference (ISC 2006), Dresden, Germany, June 2006; URL: [8]

[8] http://www.supercomp.de/isc2006/index.php?s=programs&s_nav=Conference_show&conference_id=7

[9] N. Tredennick: The Case for Reconfigurable Computing; Microprocessor Report, Vol.10 No.10, Aug 5, 1996

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

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

[12] J. Becker, M. Hübner, K. Paulsson, A. Thomas: Dynamic Reconfiguration On-Demand: Real-time Adaptivity in Next Generation Microelectronics; Proc. ReCoSoc2005, Montpellier, France, 2005

[13] E. Lübbers, M. Platzner: Reconfigurable Operating System for Reconfigurable Hardware (ReconOS); http://wwwcs.uni-paderborn.de/cs/ag-platzner/research/reconos/

[14] N. N.: MicroBlaze; http://en.wikipedia.org/wiki/Microblaze

[15] N. N.: MicroBlaze Processor Reference Guide; http://www.xilinx.com/ise/embedded/mb_ref_guide.pdf

[16] N. N.: Nios Embedded Processor; http://www.altera.com/products/ip/processors/nios/nio-index.html

[17] R. Kress et al.: A Datapath Synthesis System for the Reconfigurable Datapath Architecture; Asia and South Pacific Design Automation Conference (ASP-DAC'95), Makuhari, Chiba, Japan, Aug./Sept. 1995

[18] N. N.: XPP-III HIGH PERFORMANCE MEDIA PROCESSOR; http://www.pactxpp.com/main/index.php

[19] M. Herz et al.: Memory Organization for Data-Stream-based Reconfigurable Computing; ICECS 2002,

[20] A. DeHon. Reconfigurable Architectures for General-Purpose Computing. AI Techn. Rep. 1586, MIT, 1996

[21] R. Hartenstein: Why we need Reconfigurable Computing Education: Introduction; 1st International Workshop on Reconfigurable Computing Education (RCeducation), March 1, 2006, Karlsruhe, Germany http://xputers.informatik.uni-kl.de/staff/hartenstein/lot/HartensteinWRCE06.pdf

[22] der Begriff „von Neumann Syndrom“ wurde geprägt von Prof. C. V. Ramamoorthy, UC Berkeley

[23] A. Bode: Multicore-Architekturen; GI Informatik Spektrum, 29 5, 2006

[24] R. Hartenstein: Configware für Supercomputing: Aufbruch zum Personal Supercomputer; PIK - Praxis der Informationsverarbeitung und Kommunikation, 2006

[25] C. Trinitis: FaRM: FPGA-based Reconfigurable Microprocessors; http://www.lrr.in.tum.de/Par/arch/farm/

[26] ISLPED: http://www.islped.org/X2006/

[27] PATMOS: http://www.patmos-conf.org/

[28] N. N.: R. Associates Joins Nallatech's Growing Channel Partner Program; Companies are Using FPGAs to Reduce Costs and Increase Performance in Seismic Processing for Oil and Gas Exploration; FPGA and Structured ASIC Journal BusinessWire, August 08, 2005

[29] http://www.fpgajournal.com/news_2005/08/20050808_03.htm

[30] M. Taiji: MDGRAPE-3 chip: A 165-Gflops application-specific LSI for Molecular Dynamics Simulations; 16th IEEE Hot Chips Symposium, August 2004. URL: [31]

[31] http://www.hotchips.org/archives/hc16/3_Tue/1_HC16_Sess6_Pres1_bw.pdf

[32] Gary Smith (Gartner Dataquest): Editor’s note zu [35]

[33] A. Ast, J. Becker, et al.: Data-procedural Languages for FPL-based Machines; Proc. FPL'94, Prague, September 7-10, 1994, Lecture Notes in Computer Science, Springer, 1994

[34] J. Becker, et al.: A Novel Two-Level Hardware/Software Co-Design Framework; Journal of the Brazilian Computer Society, Dec. 1995

[35] D. Densmore, R. Passerone, A. Sangiovanni-Vincentelli: A Platform-based Taxonomy of ESL Design; Design & Test, Sept/Oct. 2006

[36] [10] N.N.: Computing Curricula 2004; Joint Task Force for Computing Curricula 2004, November 22, 2004, etc. http://www.acm.org/education/Overview_Draft_11-22-04.pdf

[37] R. Hartenstein: Overdue CS, CE and ECE Curriculum Innovations; Memo, TU-Kaiserslautern, Januar 2006; http://hartenstein.de/CurriculumInnovations.pdf

[38] Thomas S. Kuhn: The Structure of Scientific Revolutions; University of Chicago Press, 1962

7.11.06: eingereicht für das GI Informatik Spektrum

Prof. Dr. Reiner Hartenstein,
Fachbereich Informatik,
TU Kaiserslautern,
URL: http://hartenstein.de
E-mail: reiner@hartenstein.de