Reiner Hartenstein, TU Kaiserslautern, Germany

IBM

Reiner Hartenstein
TU Kaiserslautern

Reconfigurable High Performance Computing
the Next Disruptive Innovation

Apropos „reconfigurable“:
avoid confusing terminology

soft hardware
morphware®
[DARPA]
programming
morphware:
software
configure

data streams:
software
flowware

IBM Deutschland Entwicklung GmbH,
Böblingen, Germany, July 7, 2005

Programmer Education

for microelectronic systems
we also need "programmers"
but our CS curricula are obsolete:
the requirements of our labor market are ignored
this is one of the reasons of declining enrolment
young people find molecular biology more fascinating

>> The Wrong Roadmap <<

- The Wrong Roadmap
- Our Curricula are obsolete
- The overdue new Basic Model
- Coarse Grain vs. Fine Grain
- Lobbying for RC Education

Objectives

of RC are acceleration, flexibility, low cost, and, low power dissipation, mainly in:

Supercomputing, HPC
(High Performance Computing)

Embedded Systems

CV

alle akademischen Grade von der TH Karlsruhe
1959: Diplomarbeit bei Prof. Steinbuch
1969: Promotion bei Prof. Steinbuch
1974: Prof., Fakultät für Informatik, TH Karlsruhe
1977: Ordinarius*, Informatik, TU Kaiserslautern
1981: Gastprofessor CS division, UC Berkeley
2004: IEEE fellow
IBM Cell Processor

- IBM Cell Processor
- No caches - no coherence problems
- High Performance Computing (HPC)
- Embedded Systems

FPGA market

- No of design starts: + 13.4% in 2006 [Dataquest]
- Till 2010 from 80,000 to 110,000 [Dataquest]

FPGAs for Supercomputing

Bioinformatics:
- Sequence alignment, filogeny, computational analysis of DNA, protein classification, protein and RNA structure prediction and folding.
- Evolutionary Computation:
  - Genetic Algorithms, Genetic Programming, PSO and ACO applications to engineering problems;
  - Evolutionary Computation in data-mining; Metaheuristics in general.

Biomedical Engineering (BME):
- Medical Informatics, Artificial Intelligence applications in BME, simulation and pattern recognition of biomedical signals.
- See also the Bioinformatics Laboratory home-page.

FPGAs for Supercomputing

Bioinformatics:
- Motif discovery, structure discovery and classification of proteins
- Computational intelligence tools for sequence alignment, phylogeny and genomic analysis
- Data-mining in biological databases
- Evolutionary computation applied to:
  - Medical data-mining
  - Optimization problems in Engineering & Computer Science

Biomedical signal processing:
- Digital signal processing and medical instrumentation
- Biosignal pattern recognition
- Intelligent systems in medicine and health sciences:
  - Instructional systems
  - Decision-support systems
FPGA market

no of design starts:
+ 13.4% in 2006 [Dataquest]

+ 13.4% in 2006 [Dataquest]

til 2010:
from 80,000 to 110,000 [Dataquest]

N.N.: „Innovation for Prosperity“ (1)

High performance computing (HPC) has been and will continue to be a key ingredient in America’s innovation capacity. It accelerates the innovation process by shrinking “time-to-insight” and “time-to-solution” for both discovery and invention.

N.N.: „Innovation for Prosperity“ (2)

in high performance computer architecture, however, the innovation process has been massively slowed down for a long time, time-to-insight stalled for more than a decade.

N.N.: „Innovation for Prosperity“ (3)

impressive: all these respectable sponsors

the HPC Initiative: how to force the wrong road map into private economy

... understand only this parallelism solution:
the instruction-stream-based approach
the data-stream-based approach

... continuing to bang their heads against the memory wall

... continousng ignoring methodologies promising speed-ups by orders of magnitude ....

path of least resistance
avoiding a paradigm shift

Most researchers seem never to stop working on sophisticated solutions for marginal improvements ...

... continously ignoring methodologies promising speed-ups by orders of magnitude ....

... continuing to bang their heads against the memory wall

... continously ignoring methodologies promising speed-ups by orders of magnitude ....

... continuing to bang their heads against the memory wall

... continously ignoring methodologies promising speed-ups by orders of magnitude ....

... continuing to bang their heads against the memory wall

... continously ignoring methodologies promising speed-ups by orders of magnitude ....

... continuing to bang their heads against the memory wall

... continously ignoring methodologies promising speed-ups by orders of magnitude ....

... continuing to bang their heads against the memory wall

... continously ignoring methodologies promising speed-ups by orders of magnitude ....
They ignore, that Reconfigurability has become mainstream.

FPGAs: the fastest growing segment of the microelectronics market

<table>
<thead>
<tr>
<th>term</th>
<th>no. of links</th>
<th>~ 6 bio US-$</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPGA</td>
<td>~ 1,580,000</td>
<td></td>
</tr>
</tbody>
</table>

By the way ...

... the oldest and largest conference in the field:

15th International Conference on Field-Programmable Logic, Reconfigurable Computing and Applications (FPL) - FPGA

http://fpl.org
August 24 - 26, 2005, Tampere, Finland

They are going into every type of application.

Ignoring Reconfigurable Computing (RC) is the completely wrong roadmap ...

this is being changed recently by a (growing) minority in the HPC area.

massively higher performance is obtained by a fundamental paradigm shift.

Ignoring Reconfigurable Computing (RC)

\[ \text{FPGA as programmable co-processors for acceleration: Xilinx Virtex-5 Pro} \]

\[ \text{Cray provides a configware library with special algorithms for search and} \]
\[ \text{sort, DSP, and Encryption.} \]

\[ \text{in Genome-Sequenceing} \]
\[ \text{a speed-up of } > 100 \]

\[ \text{but reconfigurable logic (RL): is not computing (RC)} \]

Our Curricula are obsolete:

• The Wrong Roadmap
• Our Curricula are obsolete
• The overdue new Basic Model
• Coarse Grain vs. Fine Grain
• Lobbying for RC Education

Speech by Bill Gates at a summit meeting of US state governors:

"American high schools are obsolete."

"Our high schools - even working exactly as designed - cannot teach our kids what they need to know today."

"The high schools of today teach kids about today’s computers like on a 50-year-old mainframe.

"Our high schools were designed 50 years ago to meet the needs of another age."

"Without re-design for the needs of the 21st century, we will keep limiting - even ruining - the lives of millions of Americans every year."
The most important cultural revolution since the invention of text characters: it’s not the mainframe. 
It is the Microchip!

Speech by Bill Gates at a summit meeting of US state governors: "American high schools are obsolete."

R. H. at MSE 2005 (and earlier): “Our CS curricula are obsolete.”

Our CS departments, even working exactly as designed - cannot teach our students what they need to know today.

The Universities of today teach students about today’s Computers like on a 50-year-old mainframe.

The basic paradigm was designed 50 years ago to meet the needs of another age.

Without re-design for the needs of the 21st century, we will keep limiting - even ruining - the lives of our graduates.

Bill Gates

Joint Task Force for Computing Curricula 2004
Thomas F. LaFrenais, past VP IEEE Computer Society’s EAB, Gordon Davies (retired), U.K.’s Open University.
John Imburgia, Hofstra University.
Richard LeBlanc, Georgia Tech, Vice Chair ACM Education Board; a Team Chair for ABET’s Computing Accreditation Commission.
Barry Lunt, Brigham Young University.
Andrew McGettrick, University of Strathclyde, Glasgow.
Robert Sloan, Univ. of Illinois at Chicago, member, EAB IEEE Computer Society.
Heikki Topi, Bentley College, Waltham, MA.

Computing Curricula 2004 (2)
Within all 48 pages the term reconfigurable is not found – nor its synonyms the areas of configware and morphware are completely missing.

... how it should be morphware and configware added.

The term “embedded systems” almost ignored.

Configware Methods should be configware added.

problem space seen: how it is

CE recommendations are obsolete
Computer engineering students study the design of digital hardware systems, including computers, communications systems, and devices that contain computers. They also study software development with a focus on the software used within and between digital devices (not the software programs directly used by computer users). The emphasis of the curriculum is on hardware more than software, and it has a very strong engineering flavor.

Currently, a dominant area within computing engineering is embedded systems, the development of devices that have software components embedded in hardware. For example, devices such as cell phones, digital audio players, digital video recorders, alarm systems, x-ray machines, and laser surgical tools all require integration of hardware and embedded software, and are all the result of computer engineering.

99% of all microprocessors are used within embedded systems. The code for embedded software doubles every 10 months. Most programmers write embedded applications > 90% by the year 2010. Typical CS graduates are not qualified for this labor market.
**The overdue new Basic Model**

- The Wrong Roadmap
- Our Curricula are obsolete
- The overdue new Basic Model
- Coarse Grain vs. Fine Grain
- Lobbying for RC Education

----

### The 3rd Machine Model Became Mainstream

- **Mainframe Age**
  - 1957
  - Instruction-stream-based

- **Computer Age (PC Age)**
  - 1977
  - Hard-wired
  - Data-stream-based

- **Morphware Age**
  - 1997
  - Programmable
  - Structurally programmable

---

**Modern FPGA Bestsellers:**

The new model is reality: FPGA fabrics, together with several µprocessors, several memory banks, and other IP cores, on the same COTS microchip.

---

**Compilation: Software vs. Configware**

- **Software Engineering**
  - Source program
  - Software compiler
  - Software code

- **Configware Engineering**
  - Source program
  - Mapper
  - Configware compiler
  - Data scheduler
  - Configware code

---

**Terminology Clean-Up**

- **Programming Sources:**
  - **Configware:** for configuring morphware
  - **Flowware:** for scheduling data streams
  - **Software:** for scheduling instruction streams

Nick Tredenick's Paradigm Shifts explain the differences.

- **Software Engineering**
  - CPU resources: fixed
  - Algorithm: variable
  - 1 programming source needed

- **Configware Engineering**
  - Configware resources: variable
  - Flowware algorithm: variable
  - 2 programming sources needed
Coarse Grain vs. Fine Grain

- The Wrong Roadmap
- Our Curricula are obsolete
- The overdue new Basic Model
- Coarse Grain vs. Fine Grain
- Lobbying for RC Education

---

Coarse-grained reconfigurability

FPGAs are fine-grained by using ~ 1 bit wide CLBs (configurable logic blocks)

coarse-grained reconfigurable platforms use multi-bit wide PUs, e.g. ALU-like ...

... are much more area-efficient than FPGAs

---

Coarse grain reconfigurable

rDPU = reconfigurable datapath unit

is not a CPU

has no program counter

rDPA = reconfigurable datapath array

= array of rDPUs

---

Commercial rDPA example: PACT XPP - XPU128

---

rDPA (coarse grain) vs. FPGA (fine grain)

Status: ~1998

roughly:

- performance (MOPS/mW, orders of magnitude)
  - µProc 0
  - DSP 1
  - FPGA 2
  - rDPA 3
  - hardwired 3

roughly:

- area efficiency (transistors/chip, orders of magnitude)
  - µProc 0
  - FPGA 2
  - rDPA 4
  - hardwired 4

in special cases much higher acceleration factors!
grown awareness ...

... that the impact of morphware
means a fundamental paradigm shift
demonstrated by Google:

<table>
<thead>
<tr>
<th>term</th>
<th>no. of links</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reconfigurable</td>
<td>~ 68,400</td>
</tr>
<tr>
<td>Computing</td>
<td></td>
</tr>
</tbody>
</table>

more fascinating

Dual paradigm CS & CE curricula
including RC already for freshmen ...
... could be more fascinating
to bring enrolment up again,
provide the qualifications needed
could make the qualifications
more offshoring-resistant

Lobbying for RC education

Special interest group with IEEE Computer Society
Launch proposals to EAB, IEEE Computer Society
Push for a Special Issue of COMPUTER magazine
Meetings?
  mid' July, Massachusetts Ave, Washington, DC?
  end' August, FPL, Tampere, Finland?
other proposals?
get involved!
give me your business card
send me an e-mail

thank you

END
From Software to Configware Industry

Growing Configware Industry

Software Industry

Software Industry’s Secret of Success

Procedural personalization via RAM-based Machine Paradigm

From Software to Configware Industry

Software / Configware Co-Compilation

Juergen Becker’s CoDe-X, 1996

High level PL source

Partitioner

SW compiler / Profiler

CW compiler

SW code

CW Code

FW Code

Resource Parameters

Supporting different platforms

Growing Configware Industry

Software Industry's Secret of Success

anti machine paradigm

Repeat Success Story by a 2nd Machine Paradigm

structural personalization: RAM-based anti machine

Repeat Success Story by a 2nd Machine Paradigm!

speed-up examples from 2004 & earlier

key issue: algorithmic cleverness

<table>
<thead>
<tr>
<th>platform</th>
<th>application example</th>
<th>speed-up factor</th>
<th>method</th>
</tr>
</thead>
<tbody>
<tr>
<td>PACT Xtreme</td>
<td>16 tap FIR filter</td>
<td>x16 MOPS/mW</td>
<td>straight forward</td>
</tr>
<tr>
<td>MoM, anti</td>
<td>1-metal 1-poly nMOS***</td>
<td>&gt; x1000</td>
<td>multiple aspects</td>
</tr>
<tr>
<td>machine with</td>
<td>(computation time)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPLA* [1983]</td>
<td>grid-based DRC***</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPU 2 FPGA</td>
<td>migrate several simple application</td>
<td>x7 * x46</td>
<td>Hi level synthesis</td>
</tr>
<tr>
<td>[FPGA 2004]</td>
<td>examples</td>
<td>(compute time)</td>
<td>not spec.</td>
</tr>
<tr>
<td>DSP 2 FPGA</td>
<td>from fastest DSP</td>
<td>x100</td>
<td></td>
</tr>
<tr>
<td>[Xilinx 2004]</td>
<td>to 10 gMAGs to 1 teraMAC</td>
<td>(compute time)</td>
<td></td>
</tr>
</tbody>
</table>

*) DPLA: MPC fab via E.I.S. multi univ. project **) Design Rule Check

**) for 10-metal 3-poly CMOS expected: > x10,000

2) Wim Roelandts

Moving data around inside the Earth Simulator

Crossbar weight: 220 t, 3000 km of cable.

ES 20: TFLOPS

5120 Processors, 5000 pins each

data are moved around by software

i.e. by memory-cycle-hungry instruction streams which fully hit the memory wall

(slower than CPU clock by 2 orders of magnitude)

Moving data around inside the Earth Simulator

Crossbar weight: 220 t, 3000 km of cable.

ES 20: TFLOPS

5120 Processors, 5000 pins each

data are moved around by software

i.e. by memory-cycle-hungry instruction streams which fully hit the memory wall

(slower than CPU clock by 2 orders of magnitude)

data are moved around by software

i.e. by memory-cycle-hungry instruction streams which fully hit the memory wall

(slower than CPU clock by 2 orders of magnitude)

hypoetical branching example to illustrate time-to-space migration

S = R + (if C then A else B endif);

C = 1

simple conservative CPU example

<table>
<thead>
<tr>
<th>R</th>
<th>C</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

S = R + (if C then A else B endif);

not spec.

2) Wim Roelandts

hypoetical branching example to illustrate time-to-space migration

S = R + (if C then A else B endif);

C = 1

simple conservative CPU example

<table>
<thead>
<tr>
<th>R</th>
<th>C</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

S = R + (if C then A else B endif);

not spec.

2) Wim Roelandts

hypoetical branching example to illustrate time-to-space migration

S = R + (if C then A else B endif);

C = 1

simple conservative CPU example

<table>
<thead>
<tr>
<th>R</th>
<th>C</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

S = R + (if C then A else B endif);

not spec.

2) Wim Roelandts

hypoetical branching example to illustrate time-to-space migration

S = R + (if C then A else B endif);

C = 1

simple conservative CPU example

<table>
<thead>
<tr>
<th>R</th>
<th>C</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

S = R + (if C then A else B endif);

not spec.

2) Wim Roelandts