Data-Stream-based Reconfigurable Computing
„new“ terms
(only the terms are „new“, however, not their subject)

Software: you all know
Hardware: you all know
Morphware: structurally programmable „hardware“
Configware: sources for programming morphware

Flowware*: similar to software, but data counter manipulation:
data streams instead of instruction streams

*) no relations to „dataflow machine“ (dead area)

clean terminology and taxonomy needed for comprehensibility
flowware history:

1980: data streams 
(Kung, Leiserson)

1995: super systolic 
rDPA (Kress)

1996+: SCCC (LANL), 
SCORE, ASPRC, Bee (UCB), ...

(tutorials and courses available on all this)
### Programming: Procedural vs. Structural

<table>
<thead>
<tr>
<th>Domain</th>
<th>Procedural</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computing in ...</td>
<td>Time only*</td>
</tr>
<tr>
<td>Program source</td>
<td>Software*</td>
</tr>
<tr>
<td>„Instruction“ fetch</td>
<td>At runtime</td>
</tr>
</tbody>
</table>

**Not programmable**  
- Fully hardwired:
  - Algorithms fixed
  - Resources fixed

*) only one source needed

**Reconfigurable:**
- CPU:
  - Data-stream-based

**) Software „simulates“ flowware

---

Reconfigurable:
- Algorithms variable
- Resources variable

**Embedded systems:**
- Algorithms fixed
- Resources fixed

© 2003, reiner@hartenstein.de
Digital System Platforms clearly distinguished (1)

<table>
<thead>
<tr>
<th>Platform</th>
<th>Program Source Running on It</th>
</tr>
</thead>
<tbody>
<tr>
<td>hardware</td>
<td>(not programmable)</td>
</tr>
<tr>
<td>morphware</td>
<td></td>
</tr>
<tr>
<td>fine grain</td>
<td>rGA (FPGA)</td>
</tr>
<tr>
<td>coarse grain</td>
<td>rDPU, rDPA</td>
</tr>
<tr>
<td></td>
<td>reconfigurable data stream processor</td>
</tr>
<tr>
<td>data stream processor (hardwired)</td>
<td>flowware &amp; configware</td>
</tr>
<tr>
<td>instruction stream processor</td>
<td>flowware</td>
</tr>
</tbody>
</table>

Digital System Platforms clearly distinguished (1)

© 2003, reiner@hartenstein.de

http://hartenstein.de
more and more efforts yield only marginal improvements
areas fade away
dataflow machines are dead
shrinking supercomputing conferences
98.5% vN-only
this monopoly is dangerous

[David Padua, John Hennessy]
Dead Supercomputer Society

[Garon Bell, keynote at ISCA 2000]

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

often hardware people needed to solve CS problems
communication barriers between disciplines
not only in embedded systems: comprehensibility barrier between procedural and structural mind set
severe software quality problems
exploding design cost and implementation cost
80% of designers hate their tools...
... unusable for SW people
What are the Challenges? (1)

[ST microelectronics, MorphICs, Dataquest, eASIC]

90% by 2010

Embedd software [DTI+ law]

Communication bandwidth (Klensens law)

(1.4/year) [Moore's law]

© 2003, reiner@hartenstein.de

http://hartenstein.de
McKinsey Curve: dynamics of R&D disciplines

- CS discipline gets crusted
- Saturation: limitations met
- Challenges and motivation
- Evangelists create awareness
- Consolidation
- Innovation
- Challenges ....
- Evangelists ....
- New CS by innovation
- New discipline on top of it ....
- Year

Maturity of a discipline

© 2003, reiner@hartenstein.de
History of Computing

but awareness still missing ....

... still ignored by most CS curricula

it's already existing ...

classical CS

new CS

mainframes

technology issue and business model

data streams ...

morphware

free rider

© 2003, reiner@hartenstein.de

http://hartenstein.de
Semiconductor Revolutions

"Mainstream Silicon Application is switching every 10 Years"

Makimoto's Wave

- 1957: mainframes
- 1967: TTL
- 1977: LSI, MSI
- 1987: µproc. memory
- 1997: ASICs, accel's
- 2007: reconfigurable

Mainstream Silicon Application

- 1957: IBM
- 1967: Trittbrettfahrer
- 1977: intel
- 1987: Microsoft
- 1997: data streams...
- 2007: morphware

Technology, issue and business model

© 2003, reiner@hartenstein.de
The next EDA Industry Revolution

EDA industry paradigm switching every 7 years

courtesy [Keutzer / Newton]

1978
Transistor entry: Applicon, Calma, CV ...

1985
Schematics entry: Daisy, Mentor, Valid ...

1992
Synthesis: Cadence, Synopsys ...

1999
(Co-) Compilation
Data-Stream-based DPU arrays

2006

time of Makimoto’s 3rd wave

[Keutzer / Newton] [Hartenstein]

© 2003, reiner@hartenstein.de

http://hartenstein.de
it's time for a new CS

CS crisis: qualification problems

opportunities

next EDA wave:
high level languages

embedded systems:
hw/cw/sw co-design

configware

flowware

it's time for a new CS ...

....  a dichotomy of 2 machine paradigms

© 2003, reiner@hartenstein.de
http://hartenstein.de
The World of Matter
machine paradigm: the **Atom**

The World of Anti Matter
machine paradigm: **Anti Atom**

- Electron spinning
- Positron spinning
Matter & Antimatter of Informatics:

- DPU
- Anti Machine paradigm
- CPU
- nothing central!

instruction stream spinning ("von Neumann")

data stream spinning
The talk gives a draft of a road map toward a symbiosys of basic computing paradigms

What delays the break-through of Reconfiguable Computing?
Machine paradigms

von Neumann instruction stream machine

(instruction sequencer)

DPU

I/O

CPU

Software

Flowware

Configware

(reconf.) data-stream machine

Legend:

download

M

memory

data address generator (data sequencer)

asM*

data stream

DPU or rDPU

Legend:

download (reconf.)
heavy anti atoms: \( DPA = DPU \) array
Machine paradigms

von Neumann instruction stream machine

(instruction sequencer)

Software

Flowware

Configware

(reconf.) data-stream machine

(instruction sequencer)

Legend:
download

 memoria

data address generator
(data sequencer)

(r)DPU or rDPA

memory

(r)DPA

© 2003, reiner@hartenstein.de

http://hartenstein.de
SNN filter KressArray Mapping Example

array size: 10 x 16 = 160 rDPUs

rDPU not used used for routing only operator and routing port location marker

Legend:
- rout thru only
- not used
- backbus connect
PACT XPP: Reference Module: XPU128 Co-Processor

XPP128 ALU-Array

- 2 X PACs (Cluster)
- 128 X ALU-PAEs
- 32 X 1Kbyte RAM-PAEs
- 8X I/O Elements

[Juergen Becker, Univ. Karlsruhe]

© 2003, reiner@hartenstein.de

ALU - PAE

- PAE Core is 32- or 24-Bit ALU with DSP-Instruction Set and Controller
- Connections: Inputs + Outputs (Channels) + Events

© 2003, reiner@hartenstein.de
Throughput vs. Flexibility

- Hardwired
- FPGAs (reconfigurable logic)
- rDPAs (reconfigurable computing)
- Instruction set processors
- Standard microprocessor
- DSP

coarse grain goes far beyond bridging the gap

MOPS / mW

T. Claasen et al.: ISSCC 1999
*) R. Hartenstein: ISIS 1997

flexibility

throughput

hard-wired

FPGAs

Von Neumann

© 2003, reiner@hartenstein.de
http://hartenstein.de
Machine paradigms

von Neumann instruction stream machine

(reconf.) data-stream machine

Legend:
- download

Software

Configware embedded memory architecture*

Memory
- data address generator (data sequencer)
- data stream

DPU or rDPU

(r)DPU

(r)DPA

Configware / Flowware Compilation

rDPA
r. Data Path Array

data streams

high level source program

wrapper

mapper

configware

flowware

data sequencer

address generator

data streams

© 2003, reiner@hartenstein.de http://hartenstein.de
Efficient Memory Communication should be directly supported by the Mapper Tools

Legend:
- sequencers
- memory ports
- application
- not used

An example by Nageldinger’s KressArray Xplorer

http://kressarray.de

© 2003, reiner@hartenstein.de
Data-Stream-based Soft Machine

Compiler
Scheduler

Memory
(data memory)

memory bank
memory bank
memory bank
memory bank
memory bank

Sequencers
(data stream generator)

"instructions"

rDPA

© 2003, reiner@hartenstein.de
The Disk Farm? or a System On a Card?

The 500GB disc card
LOTS of bandwidth
A few disks replaced by
>10s Gbytes RAM
and a processor

MicroDrive:
2006: 9 GB, 50 MB/s ?
(1.6X/yr capacity, 1.4X/yr BW)
Integrated IRAM processor
Connected via crossbar switch
growing like Moore’s law
16 Mbytes; 1.6 Gflops; 6.4 Gops
10,000+ nodes in one rack!
100/board = 1 TB; 0.16 Tflops

[Gordon Bell, Jim Gray, ISCA2000]
1946: machine paradigm (von Neumann)
1980: data streams (Kung, Leiserson)
1989: anti machine paradigm introduced
1990: anti machine implementation methodology
1990: rDPU (Rabaey)
1994: anti machine high level programming language
1995: super systolic rDPA (Kress)
1996+: SCCC (LANL), SCORE, ASPRC, Bee (UCB), ...
1997: configware / software partitioning compiler (Becker)
2000: generator for rDPA with high memory bandwidth

(tutorials and courses available on all this)
## Digital System Platforms clearly distinguished (2)

<table>
<thead>
<tr>
<th>Platform</th>
<th>Program Source Running on It</th>
<th>Machine Paradigm</th>
</tr>
</thead>
<tbody>
<tr>
<td>hardware</td>
<td>(not programmable)</td>
<td>none</td>
</tr>
<tr>
<td>morphware</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fine grain</td>
<td>rGA (FPGA)</td>
<td>configware</td>
</tr>
<tr>
<td>coarse grain</td>
<td>rDPU, rDPA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>reconfigurable data stream processor</td>
<td></td>
</tr>
<tr>
<td></td>
<td>flowware &amp; configware</td>
<td></td>
</tr>
<tr>
<td>data stream processor (hardwired)</td>
<td>flowware</td>
<td></td>
</tr>
<tr>
<td>instruction stream processor</td>
<td>software</td>
<td>von Neumann machine</td>
</tr>
</tbody>
</table>
Software Industry's Secret of Success

Procedural personalization via RAM-based Machine Paradigm

1957: TTL
1967: LSI, MSI
1977: proc., memory
1987: ASICs, accel's
1997: reconfigurable
2007:

standard

custom
Configware Industry?

configware only: will remain a niche market

qualified people are not available

standard

1957
TTL
1967
LSI, MSI
1977
μproc., memory
1987
ASICs, accel's
1997
reconfigurable
2007

Repeat Success Story by new Machine Paradigm!

structural personalization: RAM-based before run time

© 2003, reiner@hartenstein.de
http://hartenstein.de
Kaiserslautern University of Technology
33 Configware Industry
The Secret of Success: Co-Compilation

not a niche market
supporting platform-based design

High level PL source

“vN” machine paradigm

Partitioner

SW compiler
Analyzer / Profiler
CW compiler

SW code
CW Code

Resource Parameters

supporting different platforms

Microsoft is looking at it

intel could provide the platforms

© 2003, reiner@hartenstein.de
http://hartenstein.de
thank you for your patience
Appendix
for discussion
High level PL source

The Secret of Success: Co-Compilation

not a niche market
supporting platform-based design

Microsoft is looking at it

intel.

Resource Parameters

Supporting different platforms

"vN" machine paradigm

anti machine paradigm

Partitioner

Analyzer/Profiler

SW compiler

CW compiler

SW code CW Code

High level PL source should provide the platforms

© 2003, reiner@hartenstein.de
### Machine Paradigms

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>driven by:</td>
<td>Instruction streams</td>
<td>data streams (no “dataflow”)</td>
</tr>
<tr>
<td>engine principles</td>
<td>instruction sequencing</td>
<td>sequencing data streams</td>
</tr>
<tr>
<td>state register</td>
<td>single program counter</td>
<td>(multiple) data counter(s)</td>
</tr>
<tr>
<td>Communication path set-up (&quot;instruction fetch&quot;)</td>
<td>at run time</td>
<td>at load time</td>
</tr>
<tr>
<td>data path resource</td>
<td>DPU (e.g. single ALU)</td>
<td>DPU or DPA (DPU array) etc.</td>
</tr>
<tr>
<td>operation</td>
<td>sequential</td>
<td>parallel pipe network etc.</td>
</tr>
</tbody>
</table>

*) e.g. Bee project Prof. Broderson
© 2003, reiner@hartenstein.de

http://hartenstein.de
# Programming Language Paradigms

<table>
<thead>
<tr>
<th>language category</th>
<th>Computer Languages</th>
<th>Languages f. Anti Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>both deterministic</td>
<td>procedural sequencing: traceable, checkpointable</td>
<td>read next data item, goto (data addr.), jump (to data addr.), data loop, loop nesting, parallel loops, escapes, data stream branching</td>
</tr>
<tr>
<td>operation sequence driven by:</td>
<td>read next instruction, goto (instr. addr.), jump (to instr. addr.), instr. loop, loop nesting no parallel loops, escapes, instruction stream branching</td>
<td>read next data item, goto (data addr.), jump (to data addr.), data loop, loop nesting, parallel loops, escapes, data stream branching</td>
</tr>
<tr>
<td>state register</td>
<td>program counter</td>
<td>data counter(s)</td>
</tr>
<tr>
<td>address computation</td>
<td>massive memory cycle overhead</td>
<td>overhead avoided</td>
</tr>
<tr>
<td>Instruction fetch</td>
<td>memory cycle overhead</td>
<td>overhead avoided</td>
</tr>
<tr>
<td>parallel memory bank access</td>
<td>interleaving only</td>
<td>no restrictions</td>
</tr>
</tbody>
</table>
Jürgen Becker’s Co-DE-X Co-Compiler
supporting platform-based design

Co-DE-X

X-C

X-C is C language extended by MoPL

Loop Transformations

Computer machine paradigm

Partitioner

Analyzer / Profiler

X-C compiler

DPSS

Resource Parameters

supporting different platforms

Host Software

KressArray Configware

Software

GNU C compiler

Xputer machine paradigm
KressArray Family generic Fabrics: a few examples

Select mode, number, width of NNports

rDPU

Select Function Repertory

rout-through and function

rout-through only

more NNports: rich Rout Resources

Examples of 2nd Level Interconnect: layouted over rDPU cell - no separate routing areas!

http://kressarray.de

http://hartenstein.de
Impact of Makimoto’s wave

Software Industry’s Secret of Success

Repeat Success Story by new Machine Paradigm!

Personalization (CAD) before fabrication

Procedural personalization via RAM-based Machine Paradigm

structural personalization: RAM-based before run time

1957
1967
1977
1987
1997
2007

TTL
LSI, MSI
μproc., memory
ASICs, accel’s
reconfigurable
standard
custom

Configware Industry

Software Industry’s Secret of Success

Repeat Success Story by new Machine Paradigm!

Personalization (CAD) before fabrication

Procedural personalization via RAM-based Machine Paradigm

structural personalization: RAM-based before run time

1957
1967
1977
1987
1997
2007

TTL
LSI, MSI
μproc., memory
ASICs, accel’s
reconfigurable
standard
custom

Configware Industry
The Dominance of the Submarine Model ...

... indicates, that our CS education system produces zillions of mentally disabled Persons

... completely disabled to cope with solutions other than software only

It's time to attack the software faculty dictatorship. Get involved!
However, current CS Education …

Hardware invisible: under the surface

Brain usage: procedural-only

This model disables ...

... is based on the Submarine Model

Algorithm

Software

procedural high level Programming Language

Assembly Language

Hardware

Software Faculty Colleagues shy away from the Paradigm Shift: their Brain hurts? - can’t be: this Half has been amputated
Hardware and Software as Alternatives

procedural structural

Algorithm partitioning

Hardware, Configware
Hardware, Software only
Hardware, Configware & Software

Brain Usage: both Hemispheres
Why Coarse Grain instead of FPGA?

Sources: Proc ISSCC, ICSPAT, DAC, DSPWorld

Transistors / chip

1980 1990 2000 2010

FPGA physical
FPGA logical
FPGA routed
supersystolic
memory
physical
logical

© 2003, reiner@hartenstein.de
http://hartenstein.de

redructed reconfigurability overhead by up to ~1000
much faster loading
a lot of more benefits

~ 10
~ 10 000

~ 10
~ 10 000

© 2003, reiner@hartenstein.de
Second Blossom of CS

progress in CS stalled by qualification problems in industry and academia

Communication barriers between disciplines

Exploding design and implementation cost

Not only in embedded systems: comprehensibility barrier between procedural and structural mind set
Severe software quality problems
Bad hardware / configware design tools: more than 80% of designers hate their tools
Procedural vs. structural

progress in CS stalled by qualification problems in industry and academia

like microprocessors also morphware is RAM-based - secret of success of software industry

Could configware industry repeat this success story?

Configware will remain a niche market, unless it comes along with hardware / configware / software co-design
... have to go beyond pointers, queues, and stacks
old CS lab course philosophy:
given an application: implement it by a program -/-

new CS freshman lab course environment:
Given an application:
  a) implement it by writing a program
  b) implement it as a morphware prototype
  c) Partition it into P and Q
     c.1) implement P by software
     c.2) implement Q by morphware
     c.3) implement P / Q communication interface
... have to go beyond pointers, queues, and stacks

Extend by including
algorithmic issues in software /morphware/ hardware migration
additional levels of parallelism: chaining, pipelining, systolic, super-systolic, wavefront arrays
additional data structures and storage organization: the new distributed memory discipline
... have to go beyond von Neumann,

Extend by including
nested machines, address generators
the anti machine paradigm
Extended taxonomy of platforms: procedural, structural,
   hardwired, reconfigurable, zhybrid systems
... have to go beyond von Neumann,

Extend by including
Configware / flowware compilers,
Procedural / structural co-compilers
(data-procedural) flowware languages
Semiconductor Revolutions

“Mainstream Silicon Application is switching every 10 Years”

Makimoto’s Wave

1st design crisis
1977
structured VLSI design
new breed (M&C)

software people
µproc., memory
instruction streams

hardware people

standard
TTL

custom
LSI, MSI

new breed needed

1957
1967
1987
1997
2007

reconfigurable

data streams

Communication gap: Terminology clean-up

© 2003, reiner@hartenstein.de
http://hartenstein.de
“Simulator-of-The-Year” Phenomenon

Digital Simulators in Use

- Tegas Logcap Laser
- Hogs Silos
- Hilo Cadat
- Verilog VHDL

Time

- early 1970’s
- early 1990’s
- ?????
"...Adoption of VHDL was one of the biggest mistakes in the history of design automation, causing users and EDA vendors to waste hundreds of millions of dollars..."

— Joe Costello, Cadence Design Systems, 1995
What is next after VHDL?

Has Designer-Oriented Innovation in Design Technology Stalled?

“With the exception of logic synthesis, there has been no significant innovation in the CAD area for the past decade.” … anonymous digital VLSI designer

- Industry infrastructural issues: CAD developers can’t deliver their ideas effectively - both university and corporate.
- Lack of effective “technology transfer” from designers to CAD developers: most non-captive CAD developers don’t have a deep, personal appreciation for the real problems facing designers.
- But the real problems for designers are getting even harder!
Flowware and Software

Software: instruction-stream-based – i.e. based on program counter manipulation

Flowware: data-stream-based – i.e. based on data counter manipulation

Software and lowware: like 2-eiige Zwillinge einführen
1. There is a very wide variety of architectures

2. Most papers have bad organization: to show authors' creativeness often less relevant details are stressed in a confusing mix of abstraction levels

3. Architectures are not described in terms of a common model

4. A common model is existing - but it's usually ignored

5. We need a comprehensible taxonomy of architectures
1. Reconfigurable instructions et extension
2. Reconfigurable co-processor
   2a. FPGA
   2b. Coarse grain
I omit 3: hardwired accelerators
I do not talk about reconfigurable instruction set processors

M&C structured VLSI design: max no. Of transistors within regular structures - Craig Mudge: regularity factor
→ structured Configware Design
- history & terminology
- skyrocketing requirements
- destructive von Neumann monopoly
- high mask cost
- low battery capacity
- new compilation model
- conclusions
Semiconductor Revolutions

"Mainstream Silicon Application is switching every 10 Years"

Makimoto's Wave

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1957</td>
<td>TTL</td>
</tr>
<tr>
<td>1967</td>
<td>LSI, MSI</td>
</tr>
<tr>
<td>1977</td>
<td>µproc., memory</td>
</tr>
<tr>
<td>1987</td>
<td>ASICs, accel's</td>
</tr>
<tr>
<td>2007</td>
<td>Reconfigurable</td>
</tr>
</tbody>
</table>

- 1st design crisis: structured VLSI design
- 2nd design crisis: reconfigurable

Communication gap: Terminology clean-up

© 2003, reiner@hartenstein.de
Terminology: DPU versus CPU ...

- DPU: data path unit
- DPA: DPU array
- GA: gate array
- rDPU: reconfigurable DPU
- rDPA: reconfigurable DPA
- rGA: reconfigurable GA

- DPU is no CPU: there is nothing central-like in a DPA
flowware defines ....

... which data item at which time at which port

flowware manipulates the data counter(s) ...

... software manipulates the program counter
1980: data streams (Kung, Leiserson)

1995: super systolic rDPA (Kress)
1996+: SCCC (LANL), SCORE, ASPRC, Bee (UCB), ...

(tutorials and courses available on all this)
skyrocketing requirements

- history & terminology
- skyrocketing requirements
- destructive von Neumann monopoly
- high mask cost
- low battery capacity
- new compilation model
- conclusions
What are the Challenges? (1)

[ST microelectronics, MorphICs, Dataquest, eASIC]
Changing Models of Computing

software design

hardware/software co-design

Software (procedural)

downloading

RAM

I/O

data path

instruction sequencer

“von Neumann”

hardware spec

host

CAD

RAM

hardware/accelerator(s)

the problem with typical CS people:

- the dominance of von Neumann
- they cannot partition
- they cannot migrate

hardware people needed

http://hartenstein.de
• history & terminology
• skyrocketing requirements
• destructive von Neumann monopoly
• high mask cost
• low battery capacity
• new compilation model
• conclusions
von Neuman does not support morphware
What about CS people?

CS people

procedural programming

languages, compiler

computer architecture

TTL


LSI, MSI

μproc., memory

FPGAs

coarse grain

soft CPUs

ASICs, accel's

1957

© 2003, reiner@hartenstein.de

http://hartenstein.de
the Datenflow Machine is dead

vN Parallelism: Resignation?

98.5% vN

Interconnect Fabrics: taken over by the opposition: Reconfigurable Computing

Flag ship example: annual IEEE ISCA conference series


© 2003, reiner@hartenstein.de http://hartenstein.de

Kaiserslautern University of Technology

73

Flag ship example: annual IEEE ISCA conference series


© 2003, reiner@hartenstein.de http://hartenstein.de

Kaiserslautern University of Technology

73
There are more Levels of Parallelism

Process level

Loop Level (data-stream-based, pipe nets, etc.)

Instruction Level (VLIW etc.)

RT Level (special architectures etc.)

Logic Level (FPGAs)

ignored by typical CS people
& ignored by CS curricula
What are the Challenges? (2)

[ST microelectronics, MorphICs, Dataquest, eASIC]

90% by 2010

- Communication bandwidth [Kleinschmidt's law]
- Integration density (1.4/year) [Moore's law]
- Processor integration density (1.2/year)
- Memory bandwidth [Patterson's law] (1.07/year)

© 2003, reiner@hartenstein.de
http://hartenstein.de
Changing Models of Computing

Software design

Software (procedural)

- downloading
- RAM
- instruction sequencer

I/O

data path

"von Neumann"

Hardware/Software co-design

- downloading
- RAM
- host

Hardware

Morphware

Configware (structural)

- downloading
- RAM
- host

Software

- CAD

Hardware/Configware/Software co-design

- re-conf. accelerator(s)

CAD

RAM

I/O

data path

"von Neumann"
no von Neumann bottleneck?

typical CS people:
• how to provide more performance to these people?
• think in terms of machine models: sequencing instruction by instruction
• cannot be turned into hardware people
• new machine paradigm needed which does not have a von Neumann bottleneck
• the anti machine has no von Neumann bottleneck
• data streams instead of an instruction stream
• flowware instead of software
The new distributed memory discipline: just in time to implement the anti machine.

• history & terminology
• skyrocketing requirements
• destructive von Neumann monopoly
• high mask cost
• low battery capacity
• new compilation model
• conclusions
What are the Challenges? (3)

[ST microelectronics, MorphICs, Dataquest, eASIC]

Avoid application-specific silicon!

Department of Trade and Industry, London

*) Department of Trade and Industry, London

© 2003, reiner@hartenstein.de http://hartenstein.de
Reconfigurability:

- fine grain (FPGAs, rGAs)
- coarse grain (PACT AG, Munich)
- multi grain (e.g. by slice bundling)
• history & terminology
• skyrocketing requirements
• destructive von Neumann monopoly
• high mask cost
• low battery capacity
• new compilation model
• conclusions
What are the Challenges? (4)

[ST microelectronics, MorphICs, Dataquest, eASIC]

Battery capacity (1.03/year)

*) Department of Trade and Industry, London
Very high throughput on low power slow FPGAs may be obtained only by algorithmic cleverness - not yet taught by CS & CSE at Universities - an urgent educational problem.
• history & terminology
• skyrocketing requirements
• destructive von Neumann monopoly
• high mask cost
• low battery capacity
• new compilation model
• conclusions
What are the Challenges? (5)

[ST microelectronics, MorphICs, Dataquest, eASIC]

- New compilation techniques needed!
- Supported by a new machine paradigm

*) Department of Trade and Industry, London
• history & terminology
• skyrocketing requirements
• destructive von Neumann monopoly
• high mask cost
• low battery capacity
• new compilation model
• conclusions
No, we are not ready for the break-through, since our computing education is obsolete, because of the von Neumann monopoly.

But all ingredients are available to jazz up our CS & CSE curricula.
thank you for your patience
The Scalability Problem

The Routing congestion Problem grows with the size of the FPGA
SNN filter KressArray Mapping Example

http://kressarray.de

array size: 10 x 16 = 160 rDPUs

rout thru only

backbus connect

not used
Xplorer Plot: SNN Filter Example

http://kressarray.de

2 hor. NNports, 32 bit
3 vert. NNports, 32 bit
route-thru-only rDPU
Conclusion: all knowledge needed is available

- machine paradigm
- languages
- compilation techniques
- anti architectural resources
- sequencing methodology: hw & sw
- hw / sw partitioning methodology
- parallel memory IP core and module generator vendors
- anything else needed

courses / embedded tutorials:
full day courses:
Configware Industry has a Chance
Conclusions

- the anti machine is the way to go for massive parallelism, also data-intensive applications
- reconfigurable anti machine for high performance with short product life cycles, unstable standards
- reconfigurable for low cost low volume production
- sparepart problem: needs new infrastructures
- Giga FPGAs highly promising - only by a new design flow: configware could repeat the success of software industry
Paradigm Shifts: Nick Tredennick's view

why 2 program sources?

instruction-stream-based computing:
- algorithms variable
- resources fixed

reconfigurable computing:
- algorithms variable
- resources variable

programmable

© 2003, reiner@hartenstein.de
http://hartenstein.de
Compilation for (r)DPA of anti machine

- high level source program (software notation)
- parameters
- morphware
- DPU library
- configware
- code generators
- flowware
- wrapper
- expression tree
- mapper
- scheduler
Moore's Law is becoming a misleading predictor of future developments.
High mask cost may be avoided completely by morphware use, or, partly by GAs (ASICs).
Morphware is the only way to obtain fault-tolerant ICs.
FPGAs may provide an important benefit for world-wide services and all other after sales consequences
Terminology has been highly confusing

„Re-configurable Hardware“ ??

this „Hardware“ is not hard !

it's Morphware

We need a concise terminology: a consensus is on the way
The key is *mapping*, rather than architecture.

<table>
<thead>
<tr>
<th>array</th>
<th>applications</th>
<th>pipeline properties</th>
<th>mapping</th>
<th>scheduling (data stream formation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>systolic array</td>
<td>regular data dependencies only</td>
<td>linear only</td>
<td>uniform only</td>
<td>linear projection or algebraic synthesis</td>
</tr>
<tr>
<td>super-systolic rDPA</td>
<td>no restrictions</td>
<td>simulated annealing or P&amp;R algorithm</td>
<td>(e.g. force-directed) scheduling algorithm</td>
<td></td>
</tr>
</tbody>
</table>

*) KressArray [1995]
Efficient Memory Communication should be directly supported by the Mapper Tools

Legend:
- sequencers
- memory ports
- application
- not used

An example by Nageldinger’s KressArray Xplorer

http://kressarray.de
Stream-based Soft Machine

Compiler

Scheduler

Memory
(data memory)

memory bank
memory bank
memory bank
memory bank
memory bank
memory bank

...“instructions”

rDPA

Sequencers
(data stream generator)
JPEG zigzag scan pattern

1. SouthScan is step by [0, 1] end SouthScan;
2. NorthEastScan is loop 8 times until [*1] step by [1, -1] endloop end NorthEastScan;
3. SouthWestScan is loop 8 times until [1, *] step by [-1, 1] endloop end SouthWestScan;
4. EastScan is step by [1, 0] end EastScan;
5. HalfZigZag is EastScan loop 3 times SouthWestScan SouthScan NorthEastScan EastScan endloop end HalfZigZag;
6. goto PixMap[1,1] HalfZigZag; SouthWestScan uturn (HalfZigZag)
**Similar Programming Language Paradigms**

<table>
<thead>
<tr>
<th>language category</th>
<th>Computer Languages</th>
<th>Xputer Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>both deterministic</td>
<td>procedural sequencing: traceable, checkpointable</td>
<td></td>
</tr>
<tr>
<td>sequencing driven by:</td>
<td>read next instruction, goto (instruction addr.), jump (to instruction addr.), instruction loop, instruction loop nesting no parallel loops, instruction loop escapes, instruction stream branching</td>
<td>read next data object, goto (data addr.), jump (to data addr.), data loop, data loop nesting, parallel data loops, data loop escapes, data stream branching</td>
</tr>
</tbody>
</table>
GAG = Generic Address Generator

GAG Scheme

Limit Stepper

Address Stepper

Base Stepper

L₀

ΔA

B₀

ΔA

L

limit

© 2003, reiner@hartenstein.de

http://hartenstein.de
GAG: Address Stepper

GAG =
Generic
Address
Generator

Limit
Base
stepVector
maxStepCount
Step
Counter
End
Detect
endExec

A
\[\Delta A\]
L
\[B_0\]
 limit

A
Address

\[\Delta A\]

\[ B_0 \]

init
tag

\(+/-\)

Escape
Clause

© 2003, reiner@hartenstein.de
http://hartenstein.de
Generic Sequence Examples

a) b) c) d) e) f) g)

Limit Slider
Base Slider
GAG
Slider Operation Demo Example
What are the Challenges?

[ST microelectronics, MorphICs, Dataquest, eASIC]

- Embedded software (DTI law)
- Communication bandwidth (Hansen's law)
- Integration density (1.4/year) (Moore’s law)
- Mask and NRE cost (1.25/year)
- Processor integration density (1.2/year)
- Memory bandwidth (Patterson's law) (1.07/year)
- Battery capacity (1.03/year)

*) Department of Trade and Industry, London
What are the Challenges?

[ST microelectronics, MorphICs, Dataquest, eASIC]

Design complexity: +40%/year  
Design productivity: +15%/year  
SIA roadmap

Battery capacity (1.03/year)
Memory bandwidth [Patterson’s law] (1.07/year)
Processor integration density (1.2/year)
Mask and NRE cost (1.25/year)

Embedded software [DIT law]
Communication bandwidth [Hartenstein’s law]
Integration density (1.4/year) [Moore’s law]

(*) Department of Trade and Industry, London
• Morphware

• Changing Models by SoC Development

• New Machine Paradigm needed

• The Dichotomy of Paradigms

• Outlook

http://www.uni-kl.de
The Morphware Market

**coarse-grained:**

rDPUs: configurable functional blocks

**fine-grained:**

cLBs, rLBs: configurable logic blocks

Top 4 PLD Manufacturers 2000

- Xilinx 42%
- Altera 37%
- Lattice 15%
- Actel 6%

Total: $3.7 Bio

PACT AG, Munich, Germany

http://pactcorp.com

- fastest growing semiconductor market segment
- PLD vendors’ and their alliances provide libraries of “soft IPs”

Configware Market
Coarse grain vs. Fine grain

Reconfigurability:

- fine grain (FPGAs, rGAs)
- coarse grain (PACT AG, Munich)
- multi grain (e.g. by slice bundling)
route-thru-only rDPU

2 hor. NNports, 32 bit
3 vert. NNports, 32 bit

http://kressarray.de

Xplorer Plot: SNN Filter Example

http://kressarray.de

2 hor. NNports, 32 bit
3 vert. NNports, 32 bit

route-thru-only rDPU

http://kressarray.de

Xplorer Plot: SNN Filter Example

http://kressarray.de
<table>
<thead>
<tr>
<th>Core</th>
<th>Architecture</th>
<th>Platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>MicroBlaze 125 MHz 70 D-MIPS</td>
<td>32 bit standard RISC 32 reg. by 32 LUT RAM-based reg.</td>
<td>Xilinx up to 100 on one FPGA</td>
</tr>
<tr>
<td>Nios 50 MHz</td>
<td>32-bit instr. set</td>
<td>Altera Mercury</td>
</tr>
<tr>
<td>Nios</td>
<td>16-bit instr. set</td>
<td>Altera 22 D-MIPS</td>
</tr>
<tr>
<td>gr1040</td>
<td>16-bit</td>
<td>Altera - Mercury</td>
</tr>
<tr>
<td>gr1050</td>
<td>32-bit</td>
<td></td>
</tr>
<tr>
<td>My80</td>
<td>i8080A</td>
<td>FLEX10K30 or EPF6016</td>
</tr>
<tr>
<td>DSPuva16</td>
<td>16 bit DSP</td>
<td>Spartan-II</td>
</tr>
<tr>
<td>Leon 25 Mhz</td>
<td>SPARC</td>
<td></td>
</tr>
<tr>
<td>ARM7 clone</td>
<td>ARM</td>
<td></td>
</tr>
<tr>
<td>uP1232 8-bit</td>
<td>CISC, 32 reg.</td>
<td>200 XC4000E CLBs</td>
</tr>
<tr>
<td>REGIS</td>
<td>8 bits Instr. + ext. ROM</td>
<td>2 XILINX 3020 LCA</td>
</tr>
<tr>
<td>Reliance-1</td>
<td>12 bit DSP</td>
<td>Lattice 4 isp30256, 4 isp1016</td>
</tr>
<tr>
<td>1Popcorn-1</td>
<td>8 bit CISC</td>
<td>Altera, Lattice, Xilinx</td>
</tr>
<tr>
<td>Acorn-1</td>
<td>1 Flex 10K20</td>
<td></td>
</tr>
<tr>
<td>YARD-1A</td>
<td>16-bit RISC, 2 opd. Instr.</td>
<td>old Xilinx FPGA Board</td>
</tr>
<tr>
<td>xr16</td>
<td>RISC integer C</td>
<td>SpartanXL</td>
</tr>
</tbody>
</table>
soft CPUs in academic teaching

- UCSC: 1990!
- Mälardalen University
- Chalmers University
- Cornell University
- Gray Research
- Georgia Tech
- Hiroshima City Univ.
- Michigan State
- Univ. de Valladolid
- Virginia Tech
- Washington U. St. Louis
- New Mexico Tech
- UC Riverside
- Tokai University
New Machine Paradigm needed

- Morphware
- Changing Models by SoC Development
- New Machine Paradigm needed
- The Dichotomy of Paradigms
- Outlook

http://www.uni-kl.de
The Dichotomy of Paradigms

- Morphware
- Changing Models by SoC Development
- New Machine Paradigm needed
- The Dichotomy of Paradigms
- Outlook
• Morphware
• Changing Models by SoC Development
• New Machine Paradigm needed
• The Dichotomty of Paradigms
• Outlook
Why fine grain?

- no specific silicon: low production volume (aerospace, automotive, military, industrial controllers, et al.)
- the spare part problem
- design flow
- coming Giga-FPGA
can configware industry repeat the success story?

- RAM-based
- Compatibility
- Scalability
- Education problems
Problems of Parallelism

enormous speed-ups: factor of 3 to >10 000

Software to FPGA migration:
algorithmic cleverness missing, no education
no methodology for interconnect estimation

Software to rDPA migration
methodology only in special areas (DSP, wireless ....)

the area of parallel algorithms needs
a complete re-orientation of its scope ...

... far beyond traditional platforms
Evolution of FPGA and its design flow

[à la S. Guccione]

as soon as Giga FPGA is available
ASIC emulation

- ASIC emulation / Rapid Prototyping: to replace simulation
- Quickturn (Cadence), IKOS (Synopsys), Celaro (Mentor)
- Hours of compilation run: inefficient since netlist-based: ...
- ... ASIC emulators will become obsolete soon
- By RTR: in-circuit execution debugging instead of emulation
- New business model: upgradable morphware is the product
- Emulation for solving the spare part problem in many areas
Nasty Matter

the wrong
machine paradigm

extremely
performance
problems
power hungry
and area
inefficient

CPU

Data Path

Instruction sequencer

central
von Neumann
bottleneck

RAM

Instruction Fetch Overhead

Address Computation Overhead

reconfigurable?
Matter vs. Antimatter: CPU vs. DPU

Data Path

instruction sequencer

DPU
Data Path Unit

+ data stream

- data stream

http://hartenstein.de
CPU: RAM-based
+ simple machine paradigm
+ scalability
+ relocatability
+ compatibility
= secret of success
of software industry

RAM

Data Path

instruction sequencer

© 2003, reiner@hartenstein.de http://hartenstein.de
### Success Factors

- For configware industry is missing:
  - FPGA compatibility,
  - Fully scalable FPGA,
  - Relocatable configuration code

- rDPUs and rDPAs do much better than FPGAs

<table>
<thead>
<tr>
<th>Property</th>
<th>Instruction Stream Based</th>
<th>Data Stream Based</th>
<th>Reconfigurable</th>
<th>Hardwired</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAM-based</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>(hardwired)</td>
</tr>
<tr>
<td>Machine paradigm</td>
<td>Yes</td>
<td>Available**</td>
<td>Available</td>
<td>Available</td>
</tr>
<tr>
<td>Compatibility</td>
<td>Yes</td>
<td>Feasible**</td>
<td>Feasible</td>
<td>Feasible</td>
</tr>
<tr>
<td>Scalability</td>
<td>Yes</td>
<td>Good**</td>
<td>Good*</td>
<td>(hardwired)</td>
</tr>
<tr>
<td>Code relocatability</td>
<td>Yes</td>
<td>Good**</td>
<td>Good*</td>
<td>(hardwired)</td>
</tr>
</tbody>
</table>

- **) Mapping coarse grain onto FPGA
- *) If KressArray used

### Success of Software Industry

For the software industry:
- FPGA compatibility
- Fully scalable FPGA
- Relocatable configuration code

For DPUs and DPAs:
- Much better than FPGAs
>>> Problems with Concurrency

- The Computer Architecture Crisis
- The Impact of Reconfigurable Platforms
- The Dichotomy of Models
- Parallelism
- Conclusions

http://www.uni-kl.de
Parallelism by Concurrency

Independent instruction streams

Difficult coordination

Bus(es) or switch box

Massive run time overhead
The Dominance of Embedded Systems

- The Computer Architecture Crisis
- The Impact of Reconfigurable Platforms
- The Dichotomy of Models
- Parallelism
- Conclusions
Summary of the Anti Machine Paradigm

- anti language primitives are almost the same (slightly extended)
- anti machine execution potential is dramatically more powerful
- provides drastically more flexibility
- not always replacing von Neumann
JPEG zigzag scan pattern

EastScan is step by [1,0] end EastScan;
SouthScan is step by [0,1] end SouthScan;
NorthEastScan is loop 8 times until [*1]
step by [1,-1] endloop end NorthEastScan;
SouthWestScan is loop 8 times until [1,*]
step by [-1,1] endloop end SouthWestScan;
HalfZigZag is
EastScan loop 3 times
SouthWestScan SouthScan NorthEastScan
EastScan endloop end HalfZigZag;

goto PixMap[1,1]
HalfZigZag:
SouthWestScan
uturn (HalfZigZag)
Address Generators for Data Streams
(data streams introduced earlier in this session)

• Introduction
• Smart Address Generators
• Address Generators for Data Streams
• Customized Memory Organization
• Conclusions

http://www.uni-kl.de
2-D Generic Data Sequence Examples

a) b) c) d) e) f) g)
GAU generic address unit Scheme

Limit Slider

Address Stepper

Base Slider

all 3 are copies of the same BSU stepper circuit

© 2003, reiner@hartenstein.de

http://hartenstein.de

Published in 1990
GAG = Generic Address Generator

GAG: Address Stepper

maxStepCount

Step Counter

End Detect

donExec

limit

A

Address

ΔA

L

Limit

Base

B₀

stepVector

사회

© 2003, reiner@hartenstein.de

http://hartenstein.de
GAG Complex Sequencer Implementation

GAG
Generic Address Generator

VLIW stack
SDS
GAG

all `been published in 1990

© 2003, reiner@hartenstein.de
GAG Slider Operation Demo Example

© 2003, reiner@hartenstein.de

Kaiserslautern University of Technology

http://hartenstein.de
The microelectronics spare part problem

- Demand: several decades of availability
- ICs do not survive storage time
- Original fab line is no more existing

- e.g. car price: ~25% electronics

[Hartenstein 2002]
The microelectronics spare part problem

key problem in many application areas: medical, aerospace, automotive, other transportation, military, industrial equipment controllers, et al.

[Hartenstein 2002]

demand /years of availability

IC market volume

IC physical life expectancy /years

2 1 0.5 0.25 0.13 0.1 0.07 µ feature size

© 2003, reiner@hartenstein.de

http://hartenstein.de
Dead Supercomputer Society

[Gordon Bell, keynote at ISCA 2000]

- ACRI
- Alliant
- American Supercomputer
- Ametek
- Applied Dynamics
- Astronautics
- BBN
- CDC
- Convex
- Cray Computer
- Cray Research
- Culler-Harris
- Culler Scientific
- Cydrome
- Dana/Ardent/
- Stellar/Stardent
- DAPP
- Denelcor
- Elexsi
- ETA Systems
- Evans and Sutherland
- Computer
- Floating Point Systems
- Galaxy YH-1
- Goodyear Aerospace MPP
- Gould NPL
- Guiltech
- ICL
- Intel Scientific Computers
- International Parallel Machines
- Kendall Square Research
- Key Computer Laboratories
- MasPar
- Meiko
- Multiflow
- Myrias
- Numerix
- Prisma
- Tera
- Thinking Machines
- Saxpy
- Scientific Computer Systems
- Systems (SCS)
- Soviet Supercomputers
- Supertek
- Supercomputer Systems
- Suprenum
- Vitesse Electronics
Microelectronics is still pushing he basic models from the times of mainframe dinosaurs after >10 technology generations. The von Neumann Microprocessor is a methusela, the steam engine of the silicon age.

Computing sciences are ultra conservative to avoid saying senile, but the von Neumann Paradigm is still the dominant doctrine... still the vN Microprocessor is a methusela, the steam engine of the silicon age.

... the vN Microprocessor is a methusela, the steam engine of the silicon age.

... the vN Microprocessor is a methusela, the steam engine of the silicon age.

... the vN Microprocessor is a methusela, the steam engine of the silicon age.

... the vN Microprocessor is a methusela, the steam engine of the silicon age.

... the vN Microprocessor is a methusela, the steam engine of the silicon age.

... the vN Microprocessor is a methusela, the steam engine of the silicon age.

... the vN Microprocessor is a methusela, the steam engine of the silicon age.

... the vN Microprocessor is a methusela, the steam engine of the silicon age.

... the vN Microprocessor is a methusela, the steam engine of the silicon age.
better to go for reconfigurable platforms

- fastest growing segment of semiconductor market
- IP reuse and silicon reuse
- FPGAs are going into every type of application
Throughput vs. Flexibility

- **Throughput** vs. **Flexibility** graph showing the relationship between throughput and feature size for different types of computer systems:
  - **Hardwired**
  - **Reconfigurable logic**
  - **Instruction set processors**
  - **rDPAs (reconfigurable computing)**

- The graph illustrates the trade-off between throughput and feature size for different types of computing devices, with FPGAs and von Neumann machines highlighted.

- **The anti machine** goes far beyond bridging the gap between throughput and flexibility.

- Credits:
  - T. Claasen et al.: ISSCC 1999
  - R. Hartenstein: ISIS 1997

- **MOPS / mW**
  - Throughput is plotted on the vertical axis.
  - Feature size is plotted on the horizontal axis.

- **μ feature size**
  - The x-axis represents feature size in micrometers.

- **Graph Sources**:
  - Kaiserslautern University of Technology
  - reiner@hartenstein.de
  - http://hartenstein.de

- © 2003, reiner@hartenstein.de

- Page 149
Why coarse grain?
## Terminology

### Instruction set processor
- **DPU**: data path unit
- **rDPU**: reconfigurable DPU
- **DPA**: data path array (DPU array)
- **rDPA**: reconfigurable DPA
- **RA**: reconguriable array
- **ISP**: instruction set processor
- **AMP**: anti machine
- **rAMP**: reconfigurable AMP

### Data stream processor
- **FPGA**: field-programmable gate array
- **FPL**: field-programmable logic
- **PLD**: programmable logic device
- **CPLD**: complex PLD

### Categories of morphware
- **morphwave**: use granularity (path width)
- **configware**: (re)configurable blocks

### Digital system platforms

<table>
<thead>
<tr>
<th>Platform Category</th>
<th>Programming Source</th>
<th>Machine Paradigm</th>
</tr>
</thead>
<tbody>
<tr>
<td>hardware</td>
<td>(not programmable)</td>
<td>none</td>
</tr>
<tr>
<td>ISP</td>
<td>software</td>
<td>von Neumann</td>
</tr>
<tr>
<td>morphware</td>
<td>configware</td>
<td>FPGA: none</td>
</tr>
<tr>
<td>data stream</td>
<td>streamware</td>
<td>anti machine</td>
</tr>
<tr>
<td>processor (AMP)</td>
<td>streamware &amp;</td>
<td>configware</td>
</tr>
<tr>
<td>rAMP</td>
<td>reconfigurable</td>
<td>reconfigurable</td>
</tr>
<tr>
<td>AMP</td>
<td>AMP</td>
<td>AMP</td>
</tr>
<tr>
<td>RA</td>
<td>fine grain (~1 bit)</td>
<td>CLBs</td>
</tr>
<tr>
<td>RA</td>
<td>coarse grain (e.g. 32 bits)</td>
<td>rDPUs (e.g. ALU-like)</td>
</tr>
<tr>
<td>rDPU slices</td>
<td>multi granular: by slice bundling</td>
<td>rDPU slices (e.g. 4 bits)</td>
</tr>
</tbody>
</table>
Problems to be solved

- Configware Market
- FPGA Market
- Embedded Systems (Co-Design)
- Hardwired IP Cores on Board
- Run-Time Reconfiguration (RTR)
- Rapid Prototyping & ASIC Emulation
- Evolvable Hardware (EH)
- Academic Expertise
- ASICs dead
- Soft CPU
- HLLs
- Problems to be solved
EDA industry shift into CS mentality

[Wojciech Maly]

• patches instead of engineering
• innovation stalled many years ago
• 85% users hate their tools
• netlist-based: do not care about efficiency, ...
• ... do not care about transistor density
FPGAs Give You

- **Instant Fabrication**
  - Get to Market Fast
  - Fix ‘em quick

- **Zero NRE Charges**
  - Low Risk
  - Low Cost at good volume
The Crisis of Computing Sciences

- Computing Sciences are in a severe crisis
- Computing curricula are obsolete because of strictly enforced „procedural-only“ blinders
- Computer Architecture and related areas have lost leadership in digital system implementation
- CS ignores > 90% μprocessors in embedded systems: 10 times more programmers will write embedded applications than computer software by 2010
- A disruptive promising therapy introduced by new approaches coming with Reconfigurable Computing
Ubiquitous embedded systems

20 billion µprocessors (2001)

> 90% in embedded systems

10 times more programmers will write embedded applications than computer software by 2010

That’s where our graduates will go

Embedded systems means:

- hardware / software co-design
- configware / software co-design
- hardware / configware / software co-design
The Situation in Computing Sciences

• Computing Sciences are in a severe crisis
• New fundamentals and R&D directions are inevitable
• my mission: getting you involved
• All knowledge needed is readily available ...
• ... even from Computing Sciences
• Silicon application and EDA provide useful concepts
• Reconfigurable Computing has the remedy
the edu gap has dramatic consequences

• Key R&D scenes are drying out or dying
• because of a lack of qualified researchers
• the embedded system design crisis gets worse
• because of a lack of qualified designers
• many innovative products cannot be sold
• because of a lack of qualified customers
• the edu gap is widening dramatically
• because of a lack of qualified educators
### Super Pipe Networks

**The key is mapping, rather than architecture**

<table>
<thead>
<tr>
<th>array</th>
<th>applications</th>
<th>pipeline properties</th>
<th>mapping</th>
<th>scheduling (data stream formation)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>systolic array</strong></td>
<td>regular data dependencies only</td>
<td>linear only</td>
<td>uniform only</td>
<td>linear projection or algebraic synthesis</td>
</tr>
<tr>
<td><strong>super-systolic DPA</strong></td>
<td>no restrictions</td>
<td></td>
<td>simulated annealing or P&amp;R algorithm</td>
<td>(e.g. force-directed) scheduling algorithm</td>
</tr>
</tbody>
</table>

*) KressArray [ASP-DAC-1995]
• now the area is going mainstream: a rapidly widening audience of non-specialists gets interested ...

• severe communication gaps due to educational deficits

• not only to users: still many hardware and EDA experts ask: isn’t it just logic design on a strange platform?

• it is time to clarify and popularize fundamental aspects and to explain, that it is a fundamentally different culture
"von Neumann" Computer: the wrong Machine Paradigm

there are some differences data stream spec

Xputer: The Soft Machine Paradigm

loosely coupled by decision data bits only

Datapath Array

reconfigurable
data counters

also for hardwired
(anti machine)

Compiler
Scheduler

"instructions"

(multiple) sequencer

RAM

Compiler
Scheduler

Datapath Array

Xputer: The Soft Machine Paradigm

Computer: the wrong Machine Paradigm
"von Neumann" s
data counter(anti machine)
Semiconductor Revolutions

"Mainstream Silicon Application is switching every 10 Years"

Makimoto's Wave

“The Programmable System-on-a-Chip is the next wave”

Tredennick's Paradigm Shifts

algorithm: fixed
resources: fixed

algorithm: variable
resources: fixed

algorithm: variable
resources: variable

Published in 1989

© 2003, reiner@hartenstein.de
http://hartenstein.de
Impact of Data-stream-based...

Repeat Success Story by new Machine Paradigm!

structural personalization: hardwired before fabrication

standard
TTL
1957
1967
1977
1987
1997
2007
reconfigurable

custom
LSI, MSI
microproc., memory
ASICs, accel's

qualified people are not available

Embedded Hardware/Configware Industry
Rapidly growing CS education gap

- Our computing curricula are obsolete
- Introduction is strictly "procedural-only"
- vN-only use of terms like "computer organisation", "computer structures", "computer architecture"
- Graduates are not prepared for the real world
  - Most applications for embedded systems (>90% by 2010)
- Our graduates are unable to compete with EE graduates
- Only a few % curricula need to be changed

- My mission: getting you involved
Binding Time vs. Computing Domain

Binding time: (Set-up of Communication Channels)

- at run time
  - microprocessor
  - parallel computer
  - array processor

- at loading time
  - supersystolic arrays
  - systolic arrays

- at compile time
  - Reconfigurable Computing

- later fabrication step
  - ASICs

- before fabrication
  - full custom ICs

programming domain:
- time domain (procedural)
- time & space (hybrid)
- space domain (structural)
Why Coarse Grain instead of FPGA?

Sources: Proc ISSCC, ICSPAT, DAC, DSPWorld

- Physical vs. logical FPGA
- Reduced reconfigurability overhead by up to ~1000
- Much faster loading
- A lot of more benefits

Graph showing transistors per chip over time (1980-2010) with trends for microprocessor, FPGA (logical, physical), supersystolic, and memory.
What are the differences?

**vN* computing:**
- computing in time
- instruction fetch at run time
- procedural programming
- instruction scheduling

**Reconfigurable Computing:**
- computing in space and time
- “instruction” fetch at compile time
- structural programming
- data scheduling
- i.e. Data-stream-based
- also hardwired implementations**
- “instruction” fetch before fabrication

*) vN stands for “von Neumann”
**) e.g. Bee project Prof. Broderson

© 2003, reiner@hartenstein.de
Basics of Binding Time

“Instruction” generalized:
including complex expressions and other datapaths

strong impact on the machine paradigm!

time of “Instruction Fetch”

run time
loading time
compile time

microprocessor parallel computer
Reconfigurable Computing
Data-stream-based Parallelism

See my other talk

<table>
<thead>
<tr>
<th>ICECS 2002</th>
<th>Memory Organisation for Datastream-based Reconfigurable Computing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dubrovnik, Croatia</td>
<td>Reiner Hartenstein, University of Kaiserslautern</td>
</tr>
<tr>
<td>September 15-18, 2002</td>
<td>Michael Herz, Agilent Technologies</td>
</tr>
</tbody>
</table>

(invited paper)
Machine paradigms

von Neumann

Data-stream machine

Software

Flowware

Configware

Embedded memory architecture*
Efficient Memory Communication should be directly supported by the Mapper Tools.

Legend:
- sequencers
- memory ports
- application
- not used

http://kressarray.de
© 2003, reiner@hartenstein.de
Terminology has been highly confusing.

Department of Trade and Industry, London

Battery capacity (1.03/year)

Memory bandwidth [Patterson's law] (1.07/year)

Microprocessor integration density (1.2/year)

Integration density (1.4/year) [Moore's law]

Communication bandwidth [Hartenstein's law]

Embedded software [DTI]*

*) Department of Trade and Industry, London
Semiconductor Revolutions

"Mainstream Silicon Application is switching every 10 Years"

"The Programmable System-on-a-Chip is the next wave"

Tredennick's Paradigm Shifts

standard

hardwired

procedural programming

structural programming

algorithm: fixed
resources: fixed

algorithm: variable
resources: fixed

algorithm: variable
resources: variable

vN machine paradigm

anti machine paradigm

Published in 1989

1957

1967

1977

1987

1997

2007

© 2003, reiner@hartenstein.de

http://hartenstein.de
The anti machine has no von Neumann bottleneck.
3 different mind sets

hardware people
- TTL
- LSI, MSI
- 1957
- 1967

CS people
- µproc.
- memory
- 1977
- 1987

new breed needed
- FPGAs
- ASICs, accel's
- 1997
- 2007

Common terminology needed

© 2003, reiner@hartenstein.de
http://hartenstein.de
Throughput vs. Flexibility

The anti machine goes far beyond bridging the gap.

Graph showing throughput vs. flexibility for different types of computing devices:

- rDPAs (reconfigurable computing)
- FPGAs
- Von Neumann instruction set processors
- Reconfigurable logic
- Standard microprocessors
- Hardwired

The graph compares MOPS per mW and µ feature size, with a label indicating "T. Claasen et al.: ISSCC 1999 (*) R. Hartenstein: ISIS 1997".

© 2003, reiner@hartenstein.de

http://hartenstein.de
Programming sources

von Neumann
instruction stream
machine
hardwired only

Anti machine
data stream machine
reconfigurable
or hardwired
## Some soft CPU core examples

<table>
<thead>
<tr>
<th>core</th>
<th>architecture</th>
<th>platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>MicroBlaze 125 MHz 70 D-MIPS</td>
<td>32 bit standard RISC 32 reg. by 32 LUT RAM-based reg.</td>
<td>Xilinx up to 100 on one FPGA</td>
</tr>
<tr>
<td>Nios</td>
<td>16-bit instr. set</td>
<td>Altera Mercury</td>
</tr>
<tr>
<td>Nios 50 MHz</td>
<td>32-bit instr. set</td>
<td>Altera 22 D-MIPS</td>
</tr>
<tr>
<td>Nios</td>
<td>8 bit</td>
<td>Altera - Mercury</td>
</tr>
<tr>
<td>gr1040</td>
<td>16-bit</td>
<td></td>
</tr>
<tr>
<td>gr1050</td>
<td>32-bit</td>
<td></td>
</tr>
<tr>
<td>My80</td>
<td>i8080A</td>
<td>FLEX10K30 or EPF6016</td>
</tr>
<tr>
<td>DSPuva16</td>
<td>16-bit DSP</td>
<td>Spartan-II</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Leon</td>
<td>SPARC</td>
<td></td>
</tr>
<tr>
<td>ARM7 clone</td>
<td>ARM</td>
<td></td>
</tr>
<tr>
<td>uP1232 8-bit</td>
<td>CISC, 32 reg.</td>
<td></td>
</tr>
<tr>
<td>REGIS</td>
<td>8 bits Instr. + ext. ROM</td>
<td></td>
</tr>
<tr>
<td>Reliance-1</td>
<td>12 bit DSP</td>
<td></td>
</tr>
<tr>
<td>1Popcorn-1</td>
<td>8 bit CISC</td>
<td></td>
</tr>
<tr>
<td>Acorn-1</td>
<td>16-bit</td>
<td></td>
</tr>
<tr>
<td>YARD-1A</td>
<td>16-bit RISC, 2 opd. Instr.</td>
<td>old Xilinx FPGA Board</td>
</tr>
<tr>
<td>xr16</td>
<td>RISC integer C</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FPGA CPUs in teaching and academic research

- UCSC: 1990!
- Märaldalen University, Eskilstuna, Sweden
- Chalmers University, Göteborg, Sweden
- Cornell University
- Gray Research
- Georgia Tech
- Hiroshima City University, Japan

- Michigan State
- Universidad de Valladolid, Spain
- Virginia Tech
- Washington University, St. Louis
- New Mexico Tech
- UC Riverside
- Tokai University, Japan
Loop Transformation Examples

**Sequential processes:**

- loop 1-16
- body
- endloop

- loop 1-8
- body
- body
- endloop

- loop 1-8
- body
- endloop

- loop
- unrolling

**Resource parameter driven**

- Co-Compilation

**Host:**

- loop 1-8
- trigger
- endloop

- loop 1-4
- trigger
- endloop

- loop 1-2
- trigger
- endloop

**Reconf.array:**

- loop 1-8
- trigger
- endloop

- loop 1-4
- trigger
- endloop

- loop 1-2
- trigger
- endloop

- strip mining
However, current CS Education ...

Hardware invisible:
under the surface

Brain usage:
procedural-only

Algorithm

Software

procedural high level Programming Language

Assembly Language

Hardware

Software Faculty Colleagues shy away from the Paradigm Shift:
their Brain hurts? - can't be:
this Half has been amputated
Hardware and Software as Alternatives

Algorithm

partitioning

procedural  structural

Brain Usage: both Hemispheres

Hardware, Configware

Software

Hardware/Softfig only

Hardware/Config only
The Dominance of the Submarine Model

.. indicates, that our CS Education System produces Zillions of Mentally Disabled Persons

... completely disabled to cope with Solutions other than Software only
# Design Space Exploration Systems

<table>
<thead>
<tr>
<th>Explorer System</th>
<th>year</th>
<th>source</th>
<th>interactive</th>
<th>status evaluation</th>
<th>status generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPE</td>
<td>1991</td>
<td>[66]</td>
<td>no</td>
<td>abstract models</td>
<td>rule-based</td>
</tr>
<tr>
<td>Clio</td>
<td>1992</td>
<td>[67]</td>
<td>yes</td>
<td>prediction models</td>
<td>device generator</td>
</tr>
<tr>
<td>DIA</td>
<td>1998</td>
<td>[68]</td>
<td>yes</td>
<td>prediction from library</td>
<td>rule-based</td>
</tr>
<tr>
<td>DSE for RAW</td>
<td>1998</td>
<td>[49]</td>
<td>no</td>
<td>analytical models</td>
<td>analytical</td>
</tr>
<tr>
<td>ICOS</td>
<td>1998</td>
<td>[76]</td>
<td>no</td>
<td>fuzzy logic</td>
<td>greedy search</td>
</tr>
<tr>
<td>DSE for Multimedia</td>
<td>1999</td>
<td>[77]</td>
<td>no</td>
<td>simulation</td>
<td>branch and bound</td>
</tr>
</tbody>
</table>
Makimoto's Wave


TTL
uproc. memory
reconfigurable

classical CS

mainframes

PC

new CS

IBM

intel.

Microsoft

© 2003, reiner@hartenstein.de

http://hartenstein.de
Billion US-$ US Market [forrester]

Million Devices delivered in the U.S.

[IDC]

Consumer PC

1500 $ 1000 $ $ 10 20 20

Consumer av. resale ($)

[forrester]


© 2003, reiner@hartenstein.de

http://hartenstein.de
Tredennick’s Paradigm Shifts

- **1957**: TTL
- **1967**: LSI, MSI
- **1977**: µproc., memory
- **1987**: ASICs, accel’s
- **1997**: 2007
- **2007**: reconfigurable

<table>
<thead>
<tr>
<th>Year</th>
<th>Standard</th>
<th>Hardwired</th>
<th>Procedural Programming</th>
<th>Structural Programming</th>
</tr>
</thead>
<tbody>
<tr>
<td>1957</td>
<td>custom</td>
<td>standard</td>
<td>procedural programming</td>
<td>standard</td>
</tr>
<tr>
<td>1967</td>
<td>hardwired</td>
<td>procedural</td>
<td>procedural programming</td>
<td>procedural programming</td>
</tr>
<tr>
<td>1977</td>
<td>procedural</td>
<td>procedural</td>
<td>structural programming</td>
<td>structural programming</td>
</tr>
<tr>
<td>1987</td>
<td>procedural</td>
<td>procedural</td>
<td>structural programming</td>
<td>structural programming</td>
</tr>
<tr>
<td>1997</td>
<td>procedural</td>
<td>procedural</td>
<td>structural programming</td>
<td>structural programming</td>
</tr>
</tbody>
</table>

- **Algorithm**: fixed/varible
- **Resources**: fixed/varible

- **2 sources**
- **New machine paradigm needed**