Reiner Hartenstein, TU Kaiserslautern, Germany

http://hartenstein.de

Reiner Hartenstein, TU Kaiserslautern, Germany

Nick Tredennick's Paradigm Shifts explain the differences.

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

**Configware Engineering**
- configware: flowware
- resources: variable
- algorithm: variable
- 2 programming sources needed

---

Organic Computing?
Bio-inspired use of FPGAs

- evolvable "hardware" community:
- crossover of chromosomes
- In love with genetic algorithms: darwinistic way to fitness thru generations of populations
- inefficient, but unexpected results possible
- simulated annealing (genetic morphing) - fitness by synthesis: highly efficient

---

Co-Compiler for Hardwired Kress/Kung Machine [e.g. Brodersen]

- automatic SW / CW partitioner

---

>> Outline <<

- Reconfigurable Computing Paradox
- Von Neumann losing its dominance
- Software vs. Configware
- The dual paradigm approach
- Coarse-grained Reconfigurable Devices
- Conclusions
coarse-grained RC: high integration density

The Reconfigurable Computing Paradox

Claassen's Law + Hartenstein’s Amendment

© 2005, reiner@hartenstein.de
http://hartenstein.de

Conclusions

FPGAs may be configured like a microprocessor for C/C++ code.
An FPGA can perform a specific algorithm at very high speed.
RC is reducing cost without loss of performance and flexibility.
RC is reducing the electricity bill and the required building floor area.
Speed-up factors of up to 4 orders of magnitude have been reported.
Compared to ASICs, prototyping time is on the order of hours rather than months, with a cost less than a tenth of that for an ASIC.
Using a high-level language, the FPGA can be programmed for a wide variety of algorithms without any deep knowledge of the underlying architecture.
The personal supercomputer is near.

Conclusions (2)

We urgently need Reconfigurable Computing Education
An Update of CS curricula is overdue.

© 2005, reiner@hartenstein.de
http://hartenstein.de
Reiner Hartenstein, TU Kaiserslautern, Germany
http://hartenstein.de

Terminology clean-up

**Programming sources:**

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

Reiner Hartenstein, TU Kaiserslautern, Germany
http://hartenstein.de

Why coarse grain

- Instead of LB (~1 bit wide), use rDPU (e.g., 32 bits wide)
- Reconfigurable Data Path Unit (e.g., rALU)
- Much more area-efficient
- Much less reconfigurability overhead

Reconfigurable Devices
- Coarse-grained Reconfigurable Devices
- Data-stream-based Computing
- The contemporary Common Model
- Reconfigurable Supercomputing
- Conclusions

"data stream": an ambiguous definition

Reconfigurable Computing is not instruction-stream-based

- It's data-stream-based
- It's different from the operation of the (indeterministic) "dataflow machine"
- Other definition also from multimedia area
- Usable definition from systolic array area

Why the speed-up ...

- Although FPGA is clock slower by x 3 or even more (most know-how from high level synthesis discipline)
- Decisions without memory cycles nor clock cycles
- Most "data fetch" without memory cycle

Data moved around by software

- i.e., by memory-cycle-hungry instruction streams which fully hit the memory wall
- P&R: move locality of operation, not data! (stolen from Bob Colwell)

Replace Caches by …

… by 16 x 16 reconfigurable data path array (rDPA)
which fits on the same chip

Similarly skilled

with hardware description languages, Hardware engineers had to adopt the methodologies and techniques of software engineers - Increased softness has an impact on even our products themselves

The required skills for your respective jobs are converging (against the grain in an age of increased specialization) and you'll soon be working with (and competing against) a new generation of embedded engineers that are similarly skilled in both disciplines.

Using FPGAs
Field-programmable FPGAs

Reducing cost without loss of performance and flexibility.

It may be configured like a general flexible micro-processor executing conventional C/C++ code, and as a highly specific programmability of FPGAs distinguishes to ASICs.

An FPGA can perform a specific algorithm at very high speed. Compared to ASICs, prototyping time is in the order of hours rather than months, with a cost less than a tenth of that for an ASIC.

Using a high-level language, the FPGA can be programmed for a wide variety of algorithms without any deep knowledge of the underlying architecture.

Co-Compiler Enabling Technology

is available from academia

only a small team needed for commercial re-implementation

on the road map to the
Personal Supercomputer

Conclusions (1)
RC suffers from fragmentation into different cultures of the many application domains.

We need a unification in dealing with problems, which are shared across many different application domains.

CS is the only domain being qualified f. such an effort

Conclusions (2)
IEEE Computer Society should advocate to improve application development methodologies

and, a common educational approach useful for the wide variety of application domains

inside IEEE Computer Society, a
TC on RC should lobby for more
Conclusions (3)

make CS more fascinating
reverse the downtrend in CS enrolment
educate not only students ...
increase membership

Strategic issue for entire IEEE Computer Society

Conclusions (4)
The personal supercomputer is near, not only for the desktop, but also for a new road map to large scale supercomputing of up to now unthinkable highest performance dimensions.

IEEE-CS is needed as a translator to explain the impact to managers and to a wide public.

IEEE-CS should accept this fascinating challenge, by spearheading the paradigm shift.

RC education last week at Karlsruhe

35 submissions from
Australia, Brasil, India, USA, and throughout Europe
Attendees declared ready to work for a task force
But education is just one of several facets ...

However ....

RC education last week at Karlsruhe

However ....

Reiner Hartenstein, TU Kaiserslautern, Germany
http://hartenstein.de
reiner@hartenstein.de

Conclusions (3)

Conclusions (4)

RC education last week at Karlsruhe

Conclusions (4)

RC education last week at Karlsruhe

Conclusions (4)

Abstract. People’s eyes drooping h/t notes illustrate the pervasiveness of Reconfigurable Computing (RC) mainstream in embedded systems already for years, and now being adapted by supercomputing (Gpp, ap, etc.). From FPGA usage as accelerators, speed-up factors by up to two orders of magnitude are reported, as well as floor space requirements and electricity usage amounts reduced by one order of magnitude. About 3 orders of magnitude and more is obtained by using coarse-grained reconfigurable datapath arrays (rDPAs) available from a number of start-ups. This is astonishing, since FPAGs and rDPAs have a substantially lower clock speed than microprocessors. Algorithmic cleverness is the secret of success, based on software to configure migration mechanisms, straining away from memory-cycle-hungry instruction stream-based computing paradigms.

The main benefit of RC platforms - having replaced the use of hardwired accelerators - is their flexibility by non-procedural programmability. This also contributed to these concepts of Organic Computing, which rely on processes of evolution, self-organization, adaptation and growthlocity. The main hurdle on the way to heart-stopping non-hardware of cheap highest performance are CS-related educational deficits causing the configure/ software chain and methodology fragmentation between the different cultures of application domains. Current CS curricula do not sufficiently meet their transdisciplinary responsibility. The talk gives a survey on fundamental issues in RC and on new directions in CS-related curricula, focused on a dual paradigm organic computing approach.

Saves more than $10,000 in electricity bills per year ($/kW) .... per 64-processor 19” rack

Reducing electricity bill by an order of magnitude

Application migration [from supercomputer] resulting in performance increase up to 4 orders of magnitude

Hits the memory wall from a different direction

Conclusions (4)

Conclusions (4)

Conclusions (4)

Conclusions (4)
Conclusions

IEEE Computer Society should advocate to introduce a dual paradigm approach - away from the monopoly of the vN mind set

IEEE Computer Society should advocate a common model useful for the wide variety of application domains

Conclusions

We need a unification in dealing with problems, which are shared across many different application domains.

RC suffers from fragmentation into different cultures of the many application domains.

Each domain uses its own trick box.

We should teach the world to think outside the box.

We need a unification in dealing with problems, which are shared across many different application domains.

CS is the only domain qualified for this unification

An Archetype Common Model needed from the Configware Industry

IEEE Computer Society should advocate to introduce a dual paradigm transdisciplinary education by using Configware Engineering as the counterpart of Software Engineering by new curricula in CS and CE

for providing an integrating dual paradigm mind set supporting a unification in dealing with problems, which are shared across many different application domains - to cure severe qualification deficiencies of our graduates.