RARE: Rethinking Architecture 
Research and Education

Chuck Thacker 
(cthacker@microsoft.com) 
Microsoft Research Silicon Valley 
October, 2010
Influential D. Patterson columns:

Seven Reasons to Shave Your Head and Three Reasons Not to: The bald truth. Commun. ACM. 49, (4): 31-32 (April, 2006)

Alas, sometimes, Dave is wrong...
Points in Dave’s second CACM column:

• **Use tools and libraries**
  – “For many CS courses, a dramatic change would simply be if students first wrote a clear specification and then built software using modern tools and software components”.

• **Embrace Parallelism.**
  – It is the only road remaining today for performance improvement

• **Join the open source movement.**

• **Build your own supercomputer.**
  – Described *RAMP*, which led to my latest projects.
The problem for Computer Architecture

• As with “real” architecture, it is about *building things*.
  – The “things” must be functional, elegant, and cost-effective.

• Academic departments haven’t been able to build computers since about 1982.
  – Chip fabrication is too expensive
  – Chip design is too complex for small student teams.

• Result: Architecture research became incremental.
The RAMP idea

• Provide an FPGA-based platform for architectural research.
• Would allow small groups to design and build significant systems again.
• I was initially skeptical
  – “FPGAs aren’t big enough”
  – “Design tools aren’t up to the job”.
• I was wrong
• But I was right that this isn’t something best done by students.
  – So we offered the RAMP folks a deal.
BEE3 Subsystems
BEE3 Program

- MSR, UCB did the detailed specifications
- MSR engaged Celestica for the implementation.
  - Better than burning out grad students
  - Pros can do the job better and faster
  - Resulting board worked the first time (unprecedented for me)
- MS licensed the design to a 3rd party company (BEECube) to build, sell, and support systems.
  - ~75 shipped to date
  - MSR supplied some basic IP (DRAM controller)
- Means that both academics and industrial customers can buy them.
  - Not the case if NSF/DARPA funded
- Good for some, but a bit pricey for wide use, so...
Beehive (‘09 – present)

• An FPGA-based many-core system
  – 13 RISC cores (100 MHz)
  – 2 GB DDR2 DRAM controller
  – Display controller
  – 1 Gb Ethernet controller
  – ~6K lines of Verilog
  – Students can understand and modify it, using only the basic Xilinx tools (ISE, ChipScope). No high-powered CAD needed.
  – Implemented on an FPGA development board that is $750 to academics.

• A software tool chain
  – C compiler, assembler, linker

• A small but growing set of libraries for frequently needed things

• Licensed for academic research use
Beehive on Xilinx XUPV5

Academic price: $750
Beehive Core CPU

Registers
32 X 32

Instruction Cache
1K X 32

Read Queue
Data from Dcache

Ra overrides
Constant
Amux
Bmux
Function
Add, Sub, Logic
Shift
Link
Out

Address Queue
Address to Dcache

Write Queue
Data to Dcache
Core local IO subsystem

AQ

WQ

Multiplier
Device 1

RS232
Device 0

DCache
Device 3

Messenger
Device 4

Lock Unit
Device 5

WhichCore
RS232

Ring

To CPU
Beehive instructions

- 32-bit instructions, 32-bit registers
- \(Rw = Ra \) Function \(Rb\) Op Count
  - Function: add, sub, logic
  - Op: Shifts
- Variants for Jumps, Memory accesses
- Support for constants
Beehive Ring Interconnect

- All wires are local
- Passes through:
  - Each core
  - Display controller
  - Ethernet controller
  - DRAM controller
- “Train” contains token + contents
- Each node can modify/append to the train
Architectural Curiosities

• No coherent memory
• No byte addressing
  – We fudge this
• No protection
  – We may add this
• No VM
• No kernel mode
Beehive uses: Education

• Architecture lab courses
  – Boards are inexpensive, so every student can have one.
  – Verilog is simple enough for students to make changes, try new things.
    • Like Stanford’s NetFPGA
  – Tool chain and libraries are familiar
    • GCC, make,...
  – Initial results are promising
Beehives at MIT

Two-week IAP course in January.

Led to a full-semester course: “Multicore Systems Laboratory”, running now.

www.web.mit.edu/6.173
Beehive uses: Research

• Forget shared memory. Use message passing

• Transactional memory
  – Allows apples-to-apples comparison with Monitors/CVs
  – Gets coherence where you need it.
  – We have a working implementation, and are writing programs to understand whether it is actually a useful abstraction for programmers.

• Do we really need...
  – Coherent shared memory?
  – Interrupts?
  – VM?
  – An OS?
Beehive Non-goals

• Emulate an existing ISA
  – Modern ISAs are *not* simple
  – Can’t do direct comparisons, only A/B experiments.

• Run Linux or other extant OSes
  – Small test programs, benchmarks
  – Barreelfish is the exception

• Have high performance
  – Can’t have this with FPGAs anyway.
  – Only needs to be fast enough to run programs much faster than a simulator.
Next steps

• Port (back) to BEE3
• Use in our own research
• TM (MSR SVL)
• Barrelish (MSR Cambridge, ETH)
• Make it more widely available for academic use (email me)
Final Thoughts

• Architecture research has been in the doldrums for 20 years.
• Recent barriers will require changes in the way we architect future systems.
• We have new opportunities and new mandates for innovation in architecture.
• We need new approaches to teaching and research.