

Advances and Challenges in Reconfigurable Computing

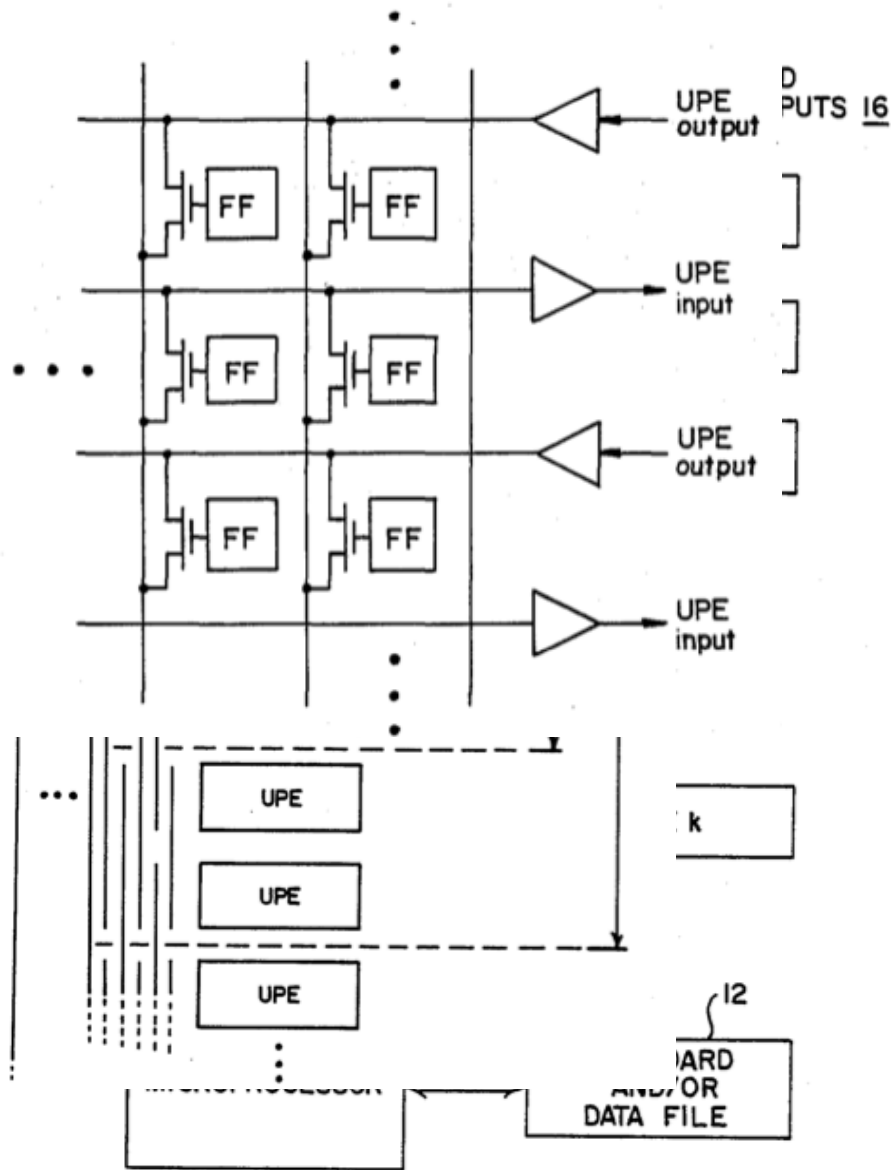


John Wawrzynek
ReConFig'2010
Dec 14

A Look at how we are
doing as a community

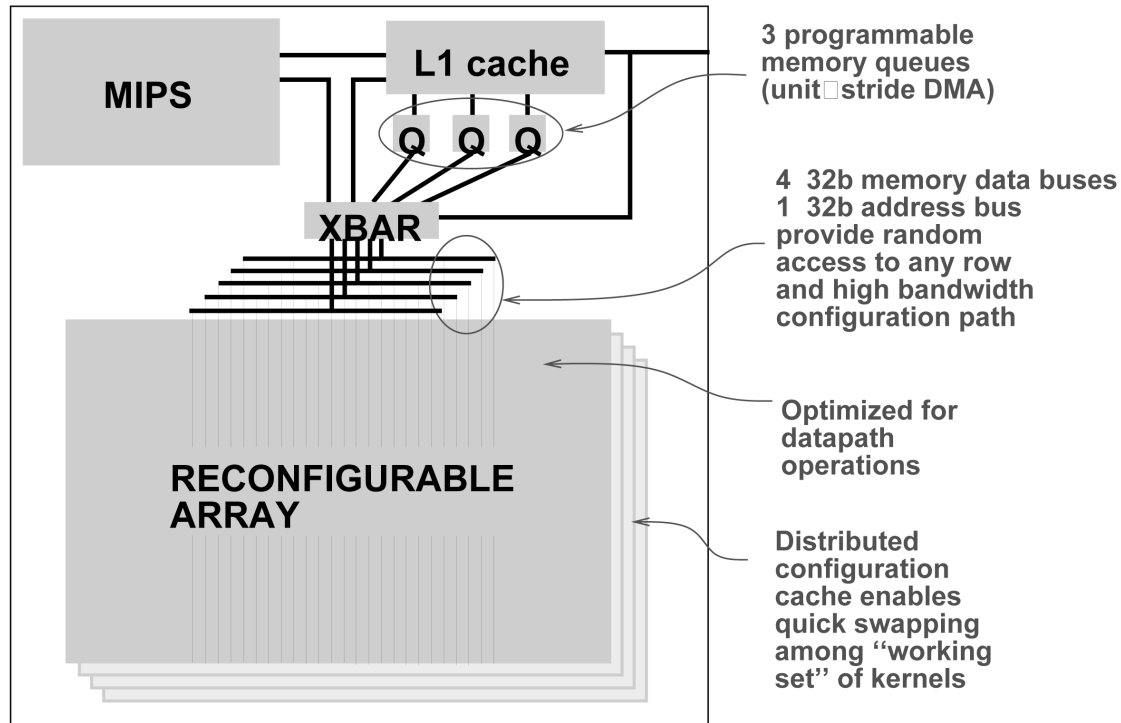


Reconfigurable Computing in 1983 (Pre FPGA)



- 4um (4000nm) nMOS, 10MHz
- Outperformed VAX750 1200x, 20 minutes down to one second (allowed realtime)
- Direct-mapped no time multiplexing of hardware
- All the key elements of FPGAs

Garp – Hybrid Processor, with C Compiler



- Pre-generated circuits for common program kernels cached within reconfigurable array and used to accelerate MIPS programs.
- nSec configuration swap time.
- Speedup – tied to single execution thread.

Function	Speedup
strlen (len 16)	1.77
strlen (len 1024)	14
sort	2.1
image median filter	26.9
DES (ECB mode)	19.6
image dithering	16.3

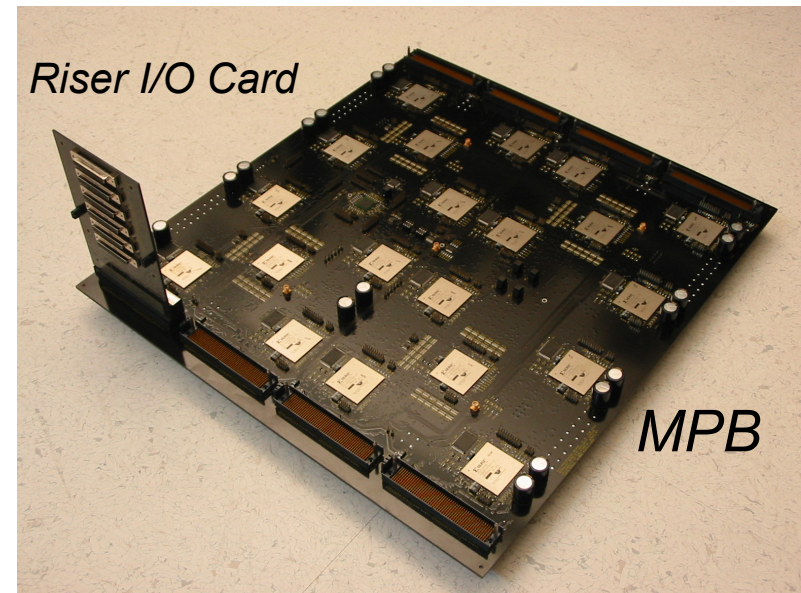
Speedups over 4-way superscalar UltraSparc on same process and comparable die size and memory system.

J. R. Hauser, J. Wawrzynek, "Garp: A MIPS Processor with a Reconfigurable Coprocessor", In *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM '97, April 16-18, 1997)*

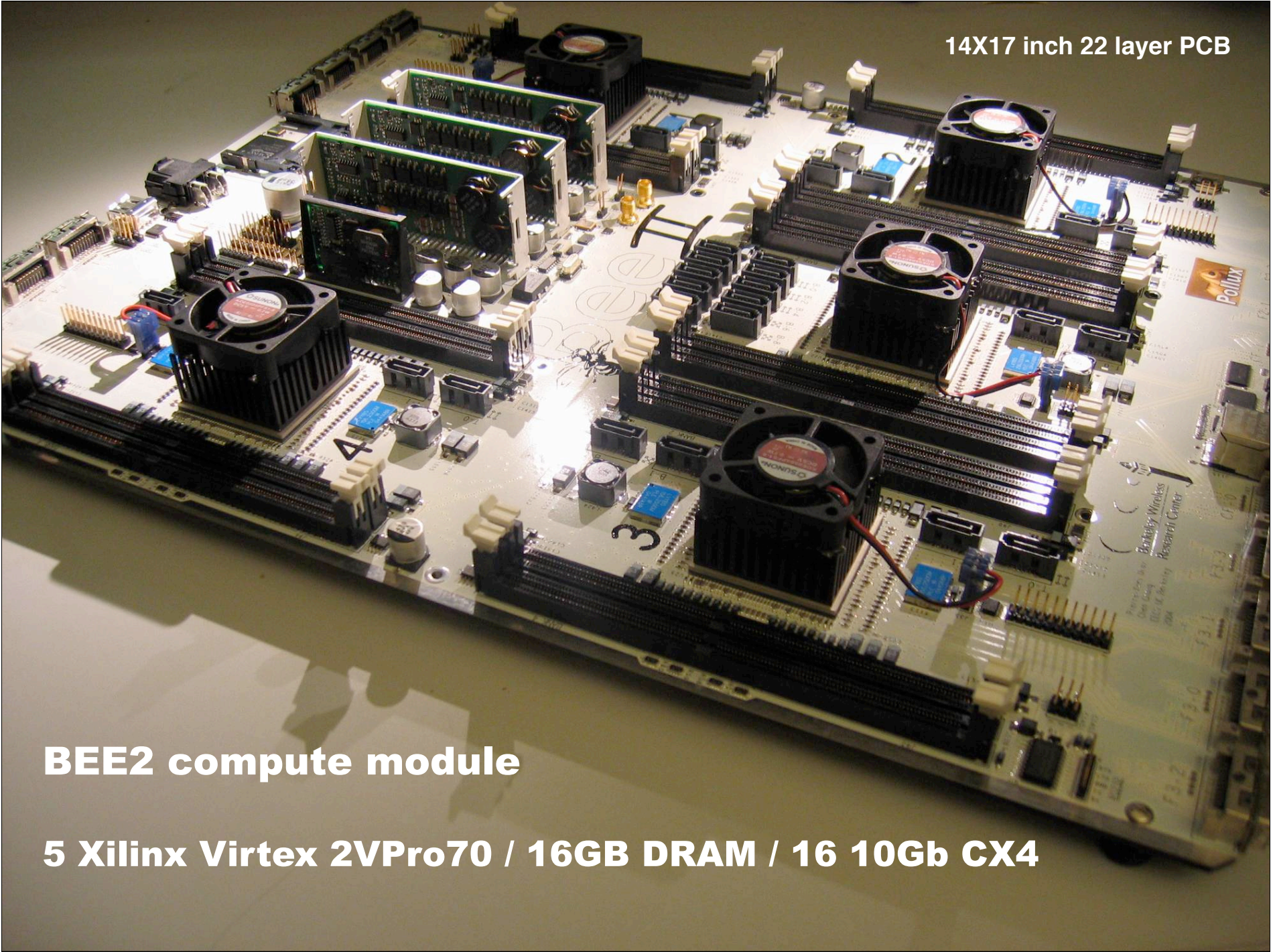
T. Callahan, J. Hauser, and J. Wawrzynek. "The Garp Architecture and C Compiler," *IEEE Computer*, April 2000.

Berkeley Emulation Engine 1 (2001)

- *Processing Board*
 - *20 Xilinx VirtexE 2000 chips, 16 on a first level mesh processing, 4 on a second level mesh.*
 - *16 ZBT SRAM chips, 1MB each.*
- *Control module (not shown)*
 - *Intel StrongARM 1110, on board 10 Base-T Ethernet, Linux OS*
- *Special built I/O cards for analog interfaces:*
 - *Ex: Radio Rx/Tx Front-Ends: 2.4 GHz transceiver, Ultrawide-band transceiver*



- Board Dimension: 53 X 58 cm
- 26 Layer PC board
- Standard 4 Mil Trace



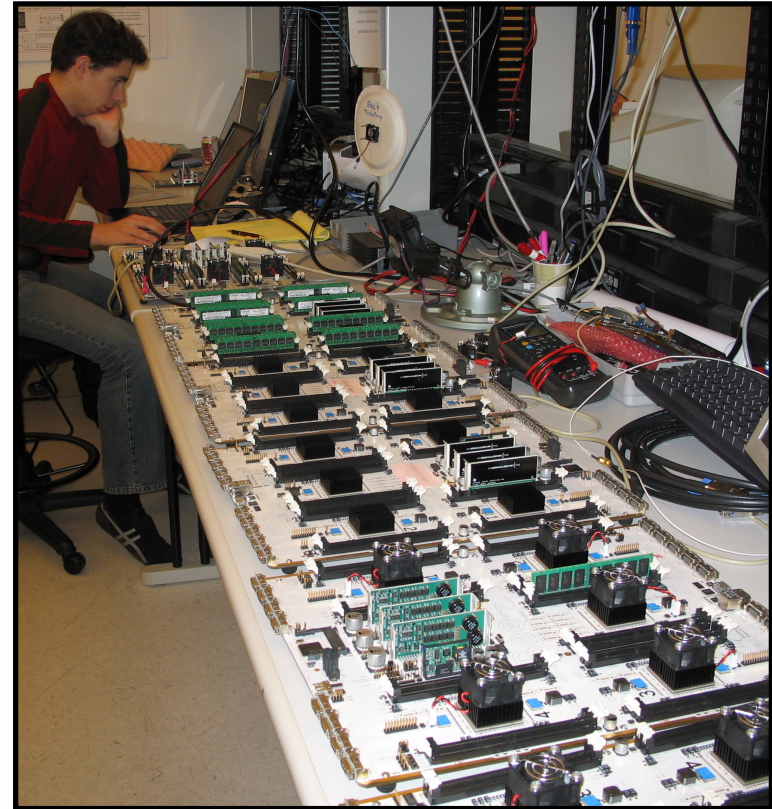
14X17 inch 22 layer PCB

BEE2 compute module

5 Xilinx Virtex 2VPro70 / 16GB DRAM / 16 10Gb CX4

Many External BEE2 Users

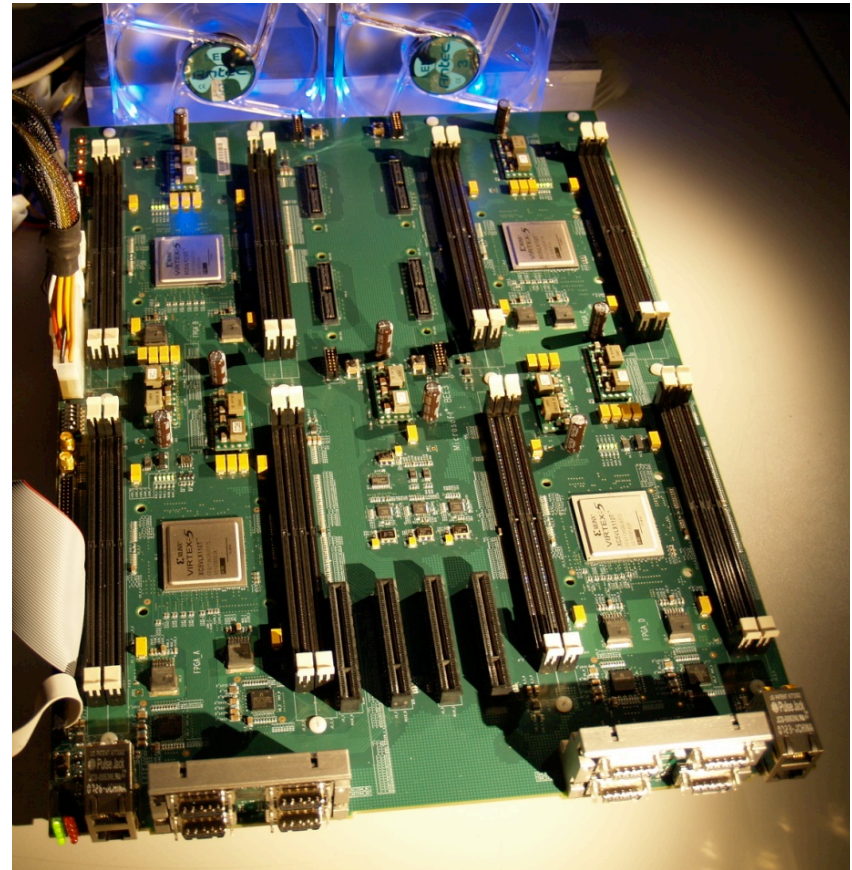
- **SSL, UC Radio Astronomy Lab:**
SETI, Allen Telescope Array
- **RAMP:** UCB, Stanford, UW, UT Austin, CMU, MIT, Intel
- **Stanford BioInformatics Group:**
Biological signaling research
- Rob Reutenbar/**CMU:** Speech Recognition
- **DARPA/DOD** Contractors
- U Toronto and Canadian Research Council
- **GSRC:** DARPA funded “BEE2” Initiative



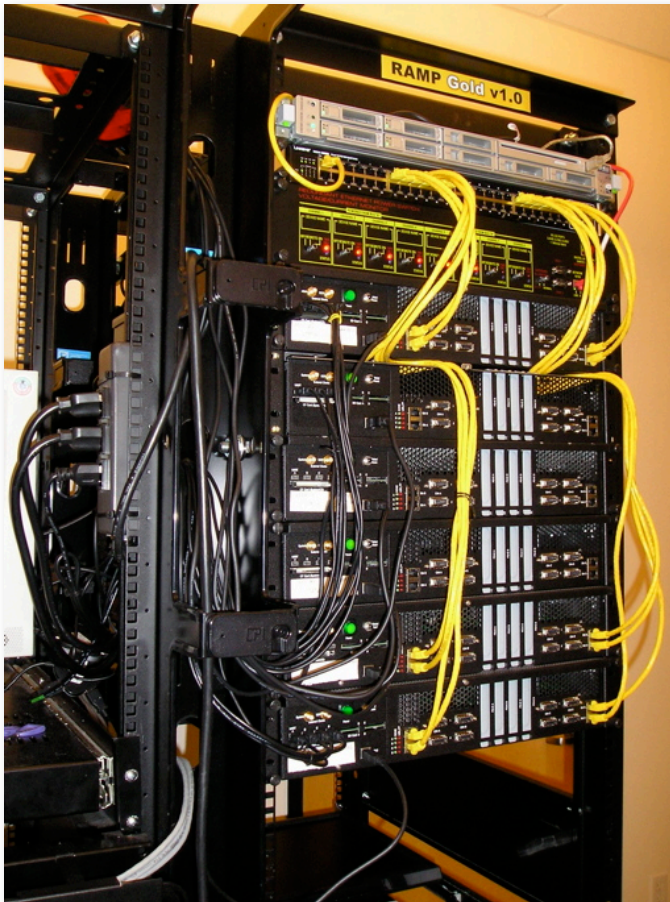
BEE3

4 Xilinx Virtex-5 FPGA, 64 GB DRAM

- 4 Xilinx Virtex-5 FPGA
- 64GB DRAM
- 160 Gpbs I/O
- 4 PCI-E x8 slots to host
- 4 high-speed GHz ADC/DAC expansion connectors
- Designed by Microsoft Research, in collaboration with UC Berkeley, Xilinx.
- Designed for handoff to commercial third party
- Licensed to BEEcube, Inc.



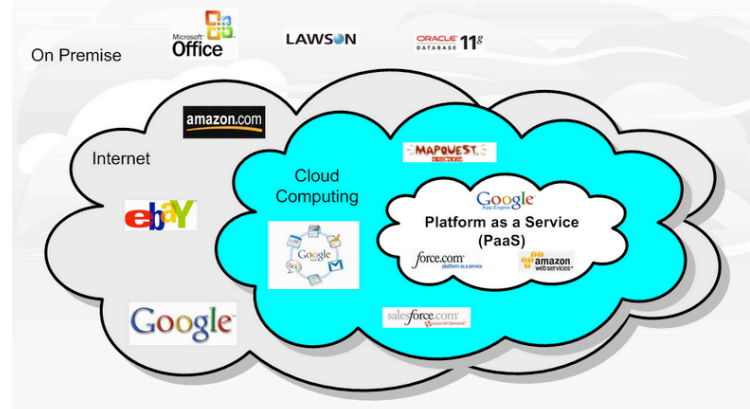
Reconfigurable Cloud-Computing



Cluster with 6 BEE3 systems.*

Still a high hurdle to acquiring, installing, maintaining FPGA platforms and development software.

Inspired by Amazon EC2 & others.



Sharing model uses common hardware, expertise, funding, and skills across entire community.

BEE3 designed as a server: rack mounting, standard GigE interfaces.

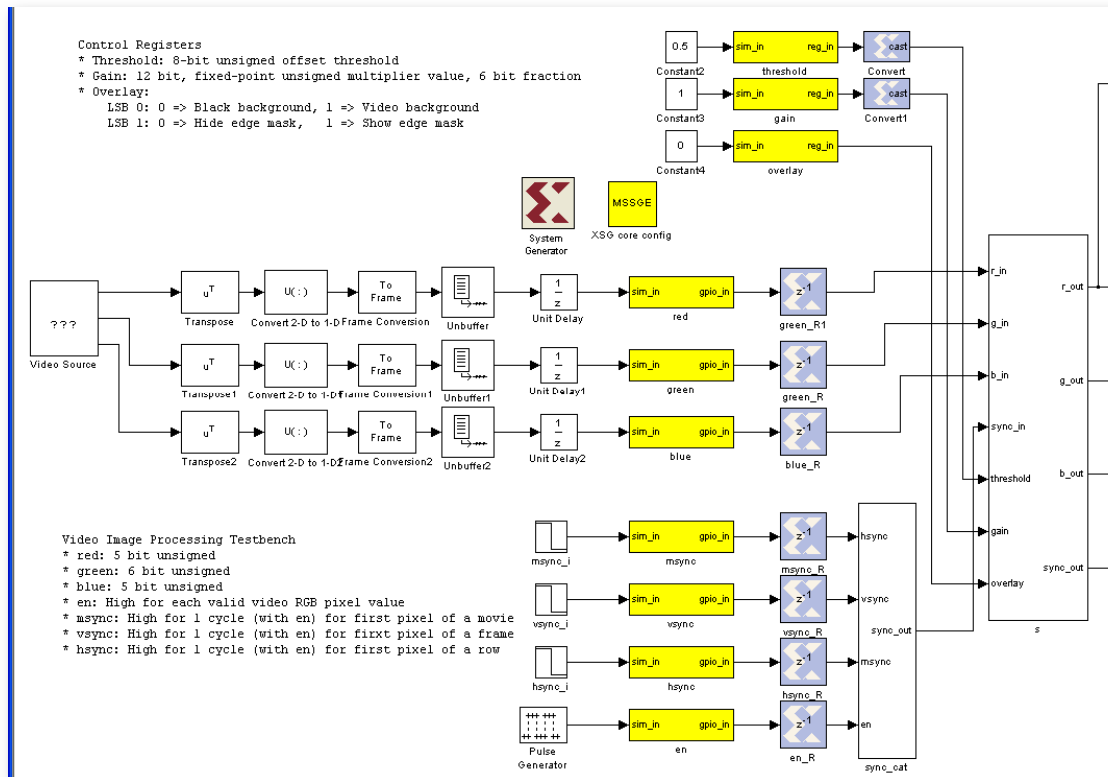
**BEE2 cluster not shown.*

BEE4 (BEEcube) : Virtex-6



Matlab/Simulink Programming Tools

Bob Brodersen, Chen Chang, Brian Richards, et. al.



- Tool flow developed by Mathworks, Xilinx, and UCB.
- User specifies design as block diagrams (for datapaths) and finite state machines for control.
- **Berkeley tools automatically map to *both* FPGAs *and* ASIC implementation.**
- BEEcube BPS support Xilinx ML boards, BEE3, and B44.

Computational Advantages of FPGAs over CPUs is well documented with Empirical Studies

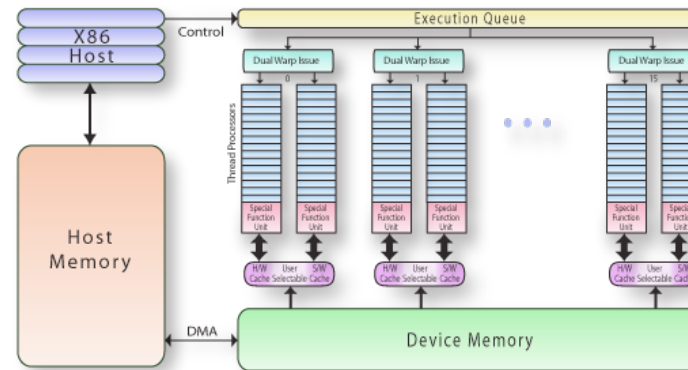
*FPGA, FPL, FCCM, ReConFig, etc. many
application areas benefit.*

*What is the source of the advantage
over conventional platforms?*

- Many of these studies derive performance advantages by simply exploiting applications parallelism. Must be careful...

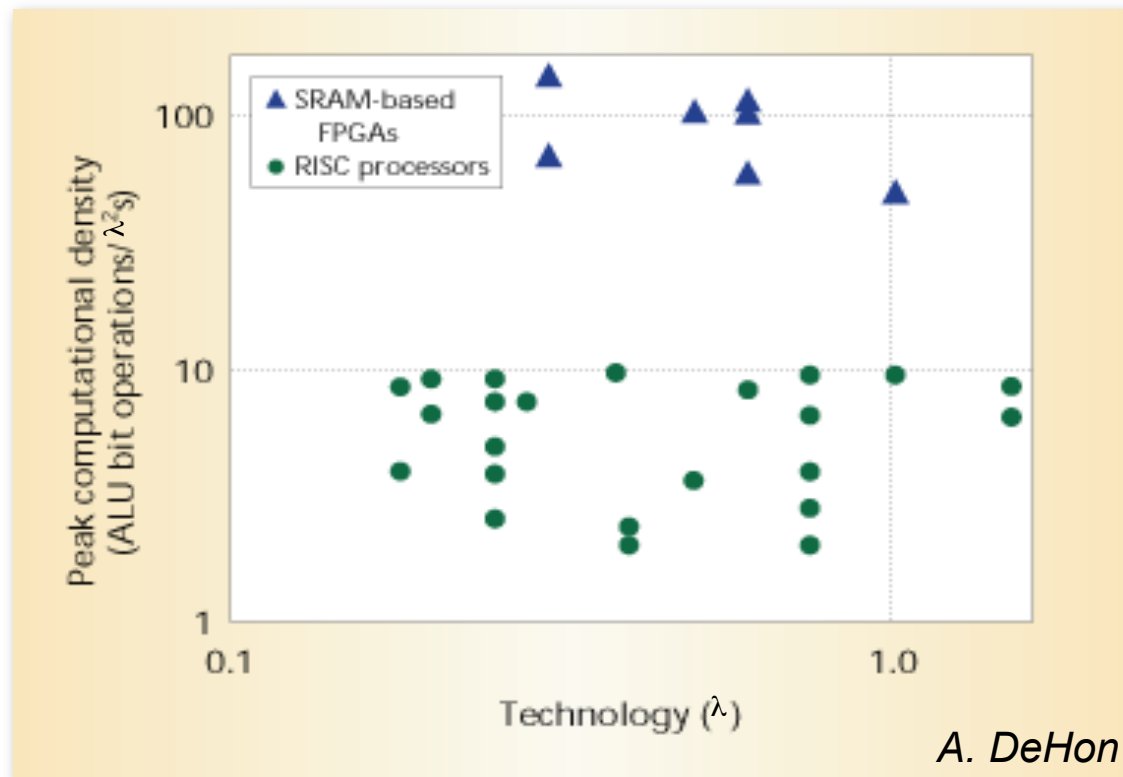
A Big Threat?

Manycore systems might eat the lunch of reconfigurable computing!



- Already we have witnessed GPGPUs systems taking over applications where RC had traditionally claimed an advantage over CPUs - SIMD/vector processing problems.
- **Response to Threat:** The RC community must continue to demonstrate when and where advantages over multicore architectures, understanding where the performance advantage comes from, with honest results.

Advantages of RC versus Processors



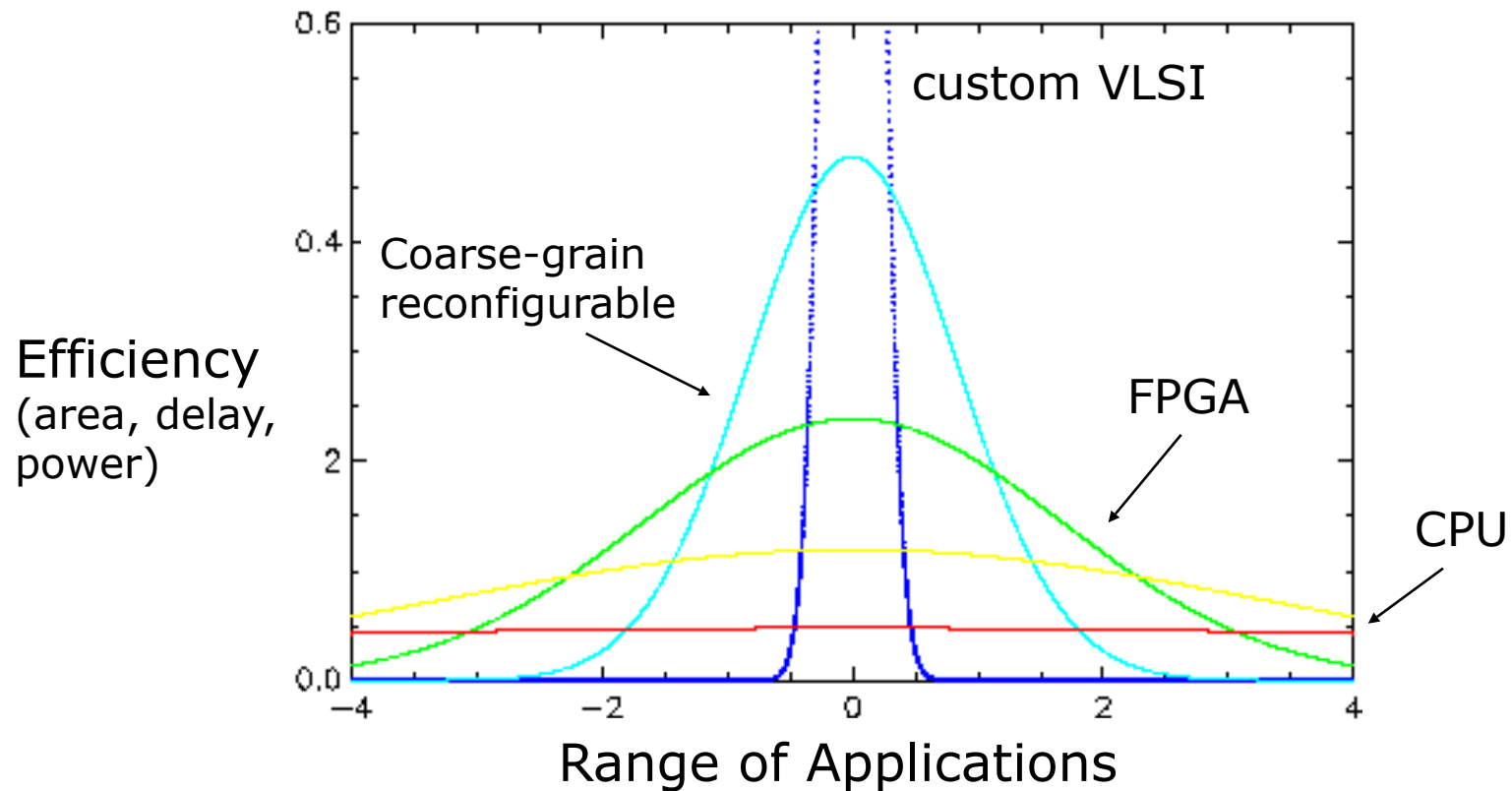
Conventional processors have several sources of inefficiency:

- Heavy time-multiplexing of Function Units (ALUs).
- Instruction issue overhead.
- Memory hierarchy to deal with memory latency.
- Operator mismatch

Peak (raw) performance

What about “yielded” performance?

FPGAs get higher performance “yield”

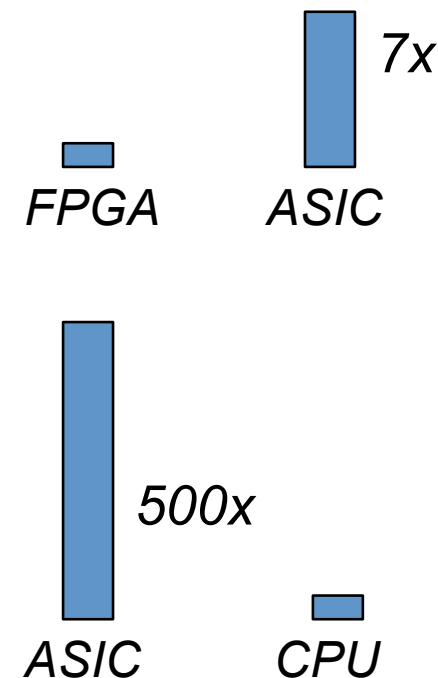


FPGA have less “flexibility overhead” than versus CPUs

Energy Efficiency

Ian Kuon and Jonathan Rose. Measuring the gap between fpgas and asics. In Proceedings of the 2006 ACM/SIGDA 14th international symposium on Field programmable gate arrays, FPGA '06, pages 21–30, New York, NY, USA, 2006. ACM

Rehan Hameed, Wajahat Qadeer, Megan Wachs, Omid Azizi, Alex Solomatnikov, Benjamin C. Lee, Stephen Richardson, Christos Kozyrakis, and Mark Horowitz. Understanding sources of inefficiency in general-purpose chips. SIGARCH Comput. Archit. News, 38:37–47, June 2010.



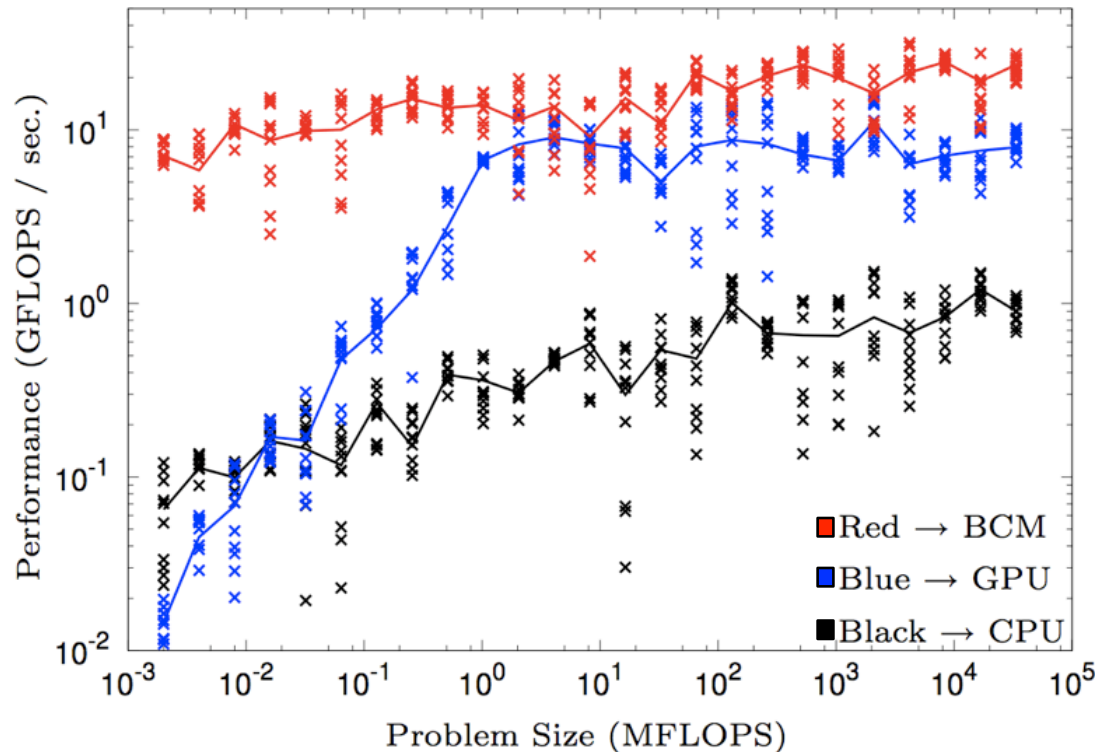
$$\therefore \text{FPGA} : \text{CPU} = 70x$$

The FPGA Efficiency Advantage

- Flexible parallelism model
- Fine-grained deployment
 - Customize datapaths/control to match application dataflow - eliminate ISA overhead
 - Customized operators (often narrow word)
 - Customized memory access with lots of memory ports.
- Relatively low flexibility overhead

Already demonstrating benefits even over FP GPUs

Bayesian Computing Machine



M. Lin, I. Lebedev, and J. Wawrzynek, "High-Throughput Bayesian Computing Machine with Reconfigurable Hardware," Proceedings of the Eighteenth ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, February 21-23, 2010.

- 16 processing node (single precision FP) machine mapped to BEE3 (Virtex-5 LX155T) FPGA
- Compared to: C code running on 2.4 GHz Core 2 Duo Intel Processor, CUDA code running on NVIDIA GeForce 9400M.
- BCM prototype throughput ~20.4 GFLOPS.

So why are FPGA not Supplanting (or at least seriously competing with) CPUs/GPUs?

- Most Common Answer: “Lack of High-Level Programming Models and Languages”
- Actually, all aspect of “Ease of Use”
 1. Multiple programming models with efficient mapping
 2. Fast compile/debug loop
 3. Standard “seamless” operating environment
 - I/O, process management, memory management, with application portability

“What do microprocessors have that lead to their ease of use and success?”

- **Standardization of Simple Hardware abstraction (ISA) and open documented binary interface.**
- Leads to portable applications, libraries, operating systems
- Leads to open source tools, OSs, academic/research community involvement
- “ISA” started in the 1960’s with the IBM 360

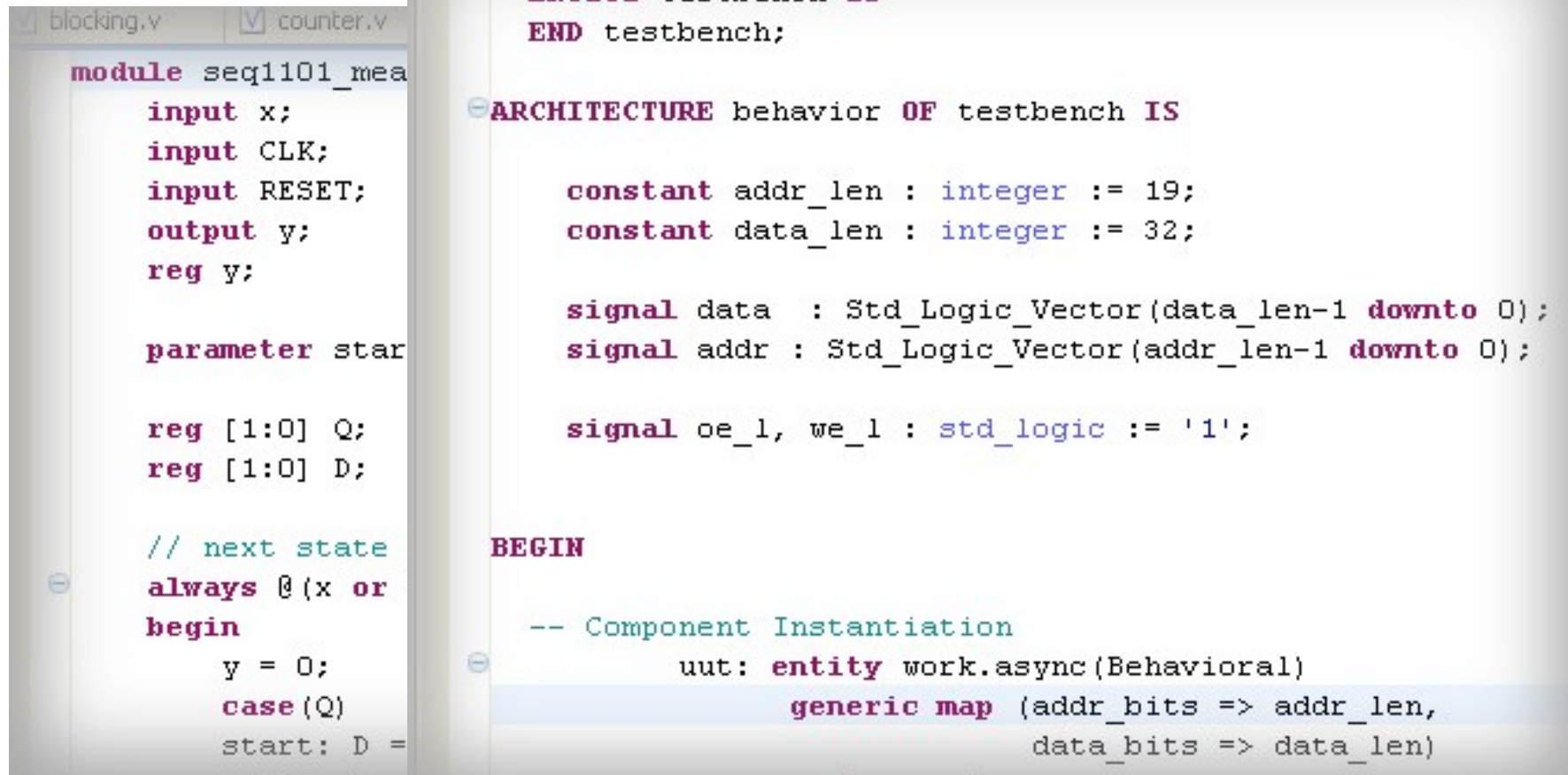
IBM System 360



So how are we doing with FPGAs?

1. Programming Models and Languages

- Most application still developed with Verilog/
VHDL



```
blocking.v  counter.v

module seq1101_mea
    input x;
    input CLK;
    input RESET;
    output y;
    reg y;

    parameter start

    reg [1:0] Q;
    reg [1:0] D;

    // next state
    always @(x or
    begin
        y = 0;
        case (Q)
        start: D =

ENTITY testbench IS
END testbench;

ARCHITECTURE behavior OF testbench IS

    constant addr_len : integer := 19;
    constant data_len : integer := 32;

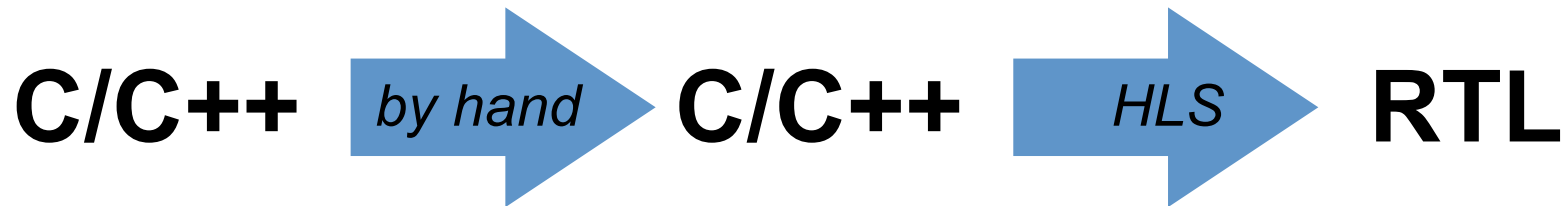
    signal data : Std_Logic_Vector(data_len-1 downto 0);
    signal addr : Std_Logic_Vector(addr_len-1 downto 0);

    signal oe_l, we_l : std_logic := '1';

BEGIN

    -- Component Instantiation
    uut: entity work.async(Behavioral)
        generic map (addr_bits => addr_len,
                     data_bits => data_len)
```


1. Programming Models and Languages



- Helps, must rewrite with HW implementation in mind, doesn't handle memory well, single thread,
- Need way to handle coarser grain parallelism

Widespread Panic in the Industry



Computing industry is betting their future on parallel processors. But how do we program multicore and manycore systems?

Dave Patterson: “Industry has already thrown the hail-mary pass. . . But nobody is running yet.”

UPCRC Partners in Research

- Intel & Microsoft provide funding and guidance
- Universities direct groundbreaking research



Prof. David Patterson
UCB UPCRC Director



Prof. Wen-Mei Hwu



Prof. Marc Snir
UIUC UPCRC Co-Directors

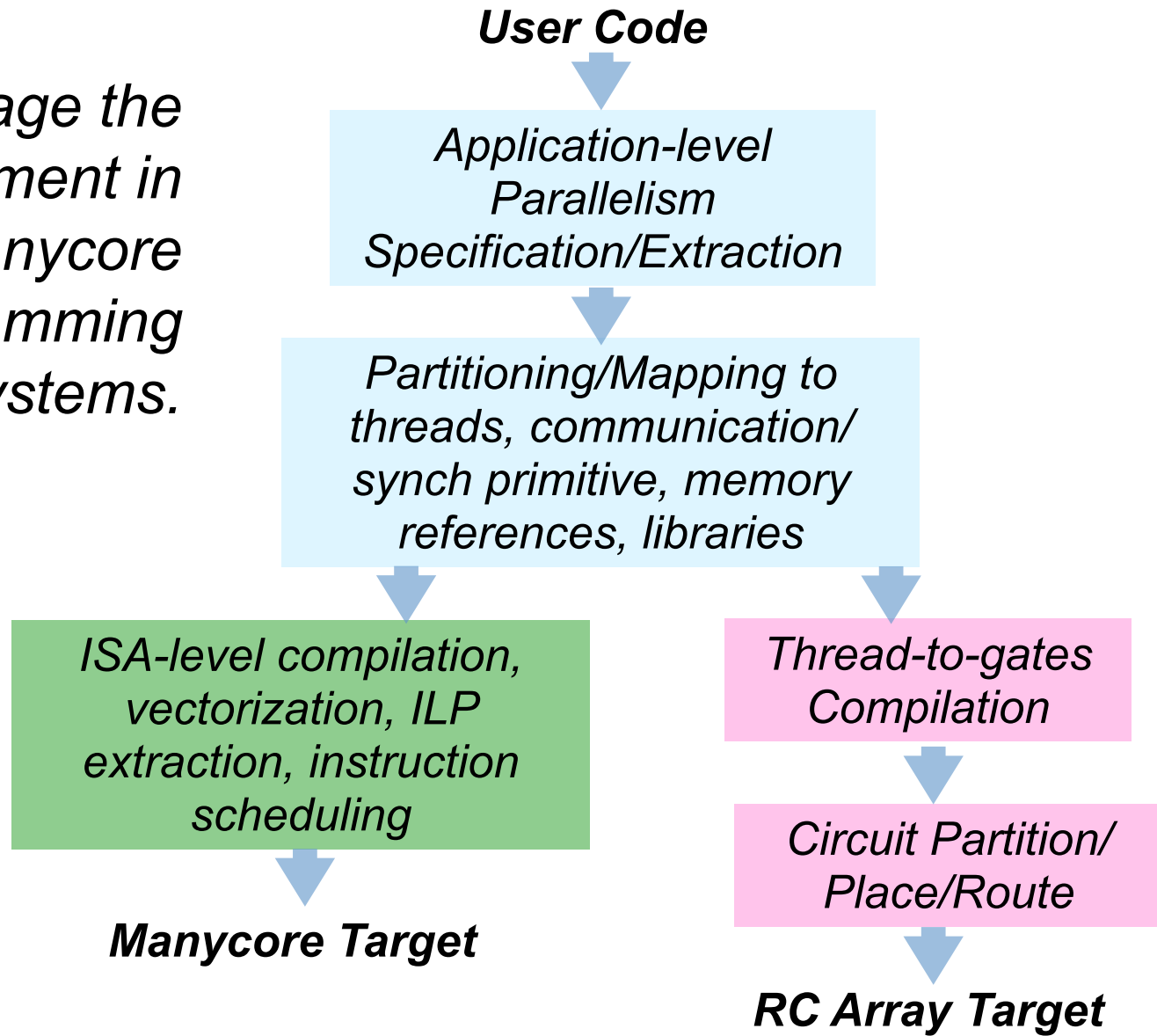
Intel / Microsoft



Now, huge investment is developing new methods for programming new parallel architectures.

Great News for Reconfigurable Computing

Leverage the investment in manycore programming systems.



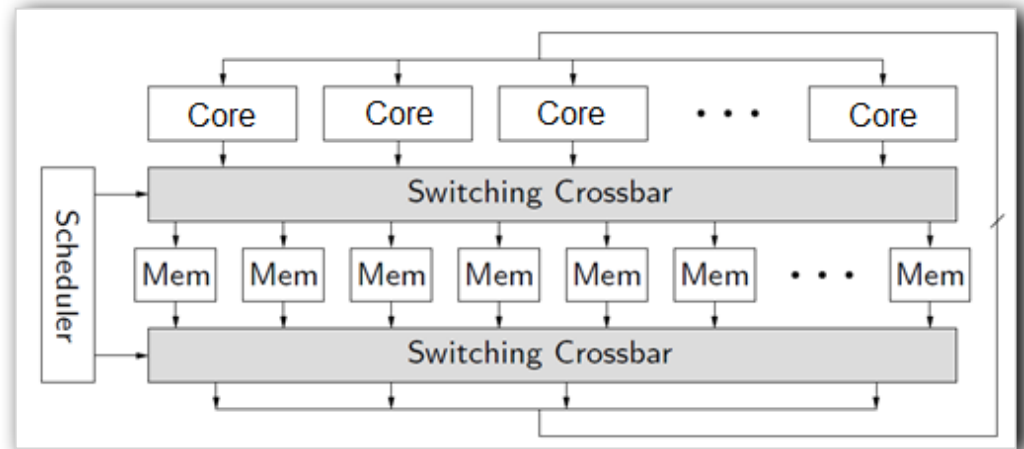
1. Programming Models and Languages

A. Papakonstantinou, K. Gururaj, J. A. Stratton, D. Chen, J. Cong and W. Hwu.

FCUDA: Enabling efficient compilation of CUDA kernels onto FPGAs. Proceedings of the 7th Symposium on Application Specific Processors, pp.35-42, July 2009. Boston, MA

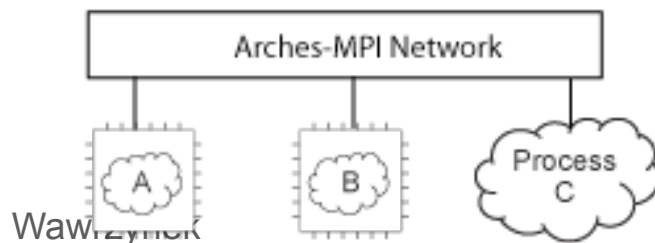
Mingjie Lin, Ilia Lebedev, and John Wawrzynek, "**OpenRCL**: Low-Power High-Performance Computing with Reconfigurable Devices ," Proceedings of the 20th International Conference on Field Programmable Logic and Applications (FPL2010), Aug. 31st - Sep. 2nd, 2010. *Short Paper*.

I. Lebedev, S. Cheng, A. Doupnik, J. Martin, C. Fletcher, D. Burke, M. Lin, & J. Wawrzynek, "**MARC**: A Many-Core Approach to Reconfigurable Computing," ReConFig 2010



Step 3 - FPGA-CPU Mix

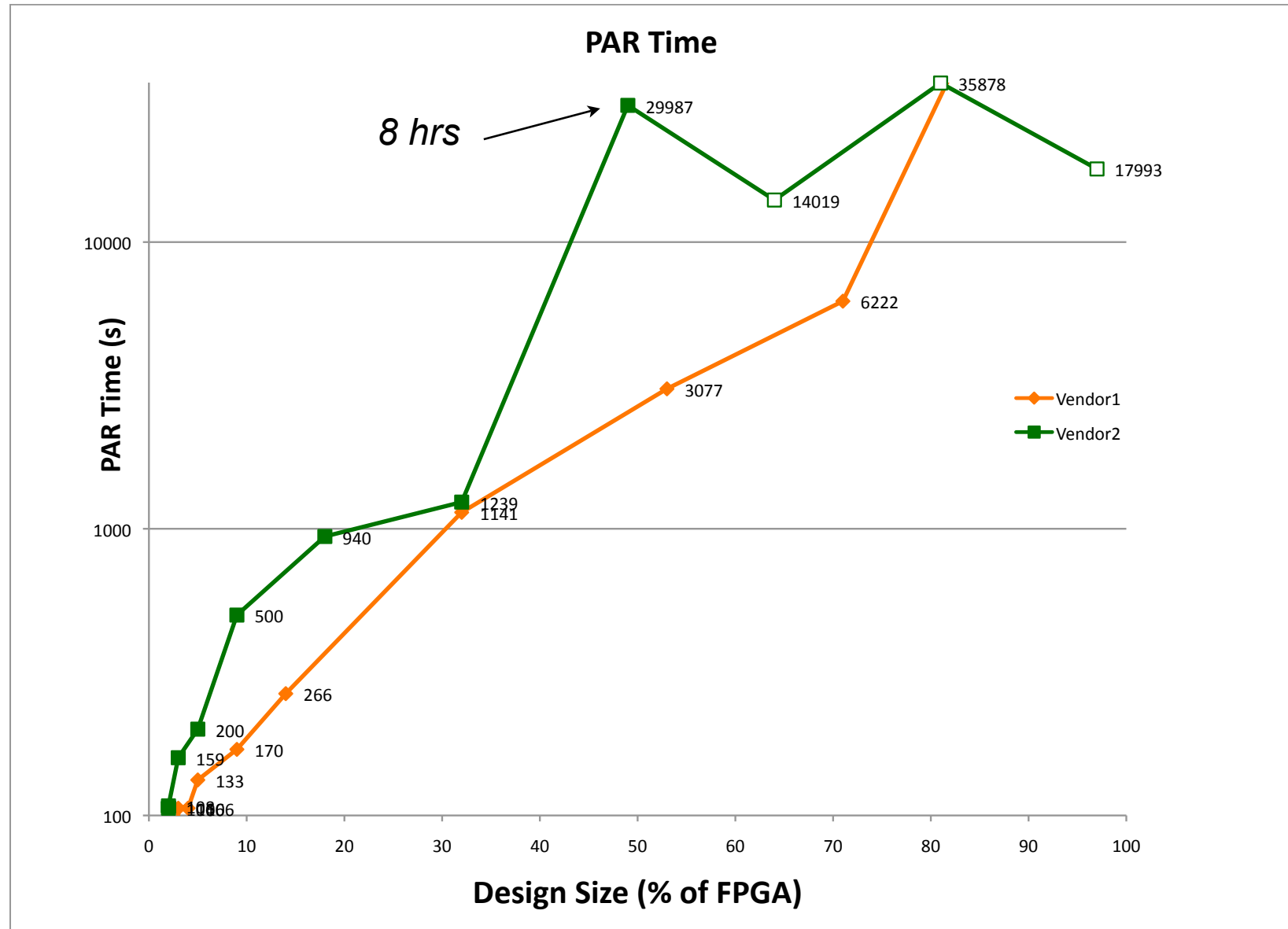
Accelerator Modeling



Arches/MPI

ReConFig 12/14/2010

2. Fast Compile / Debug

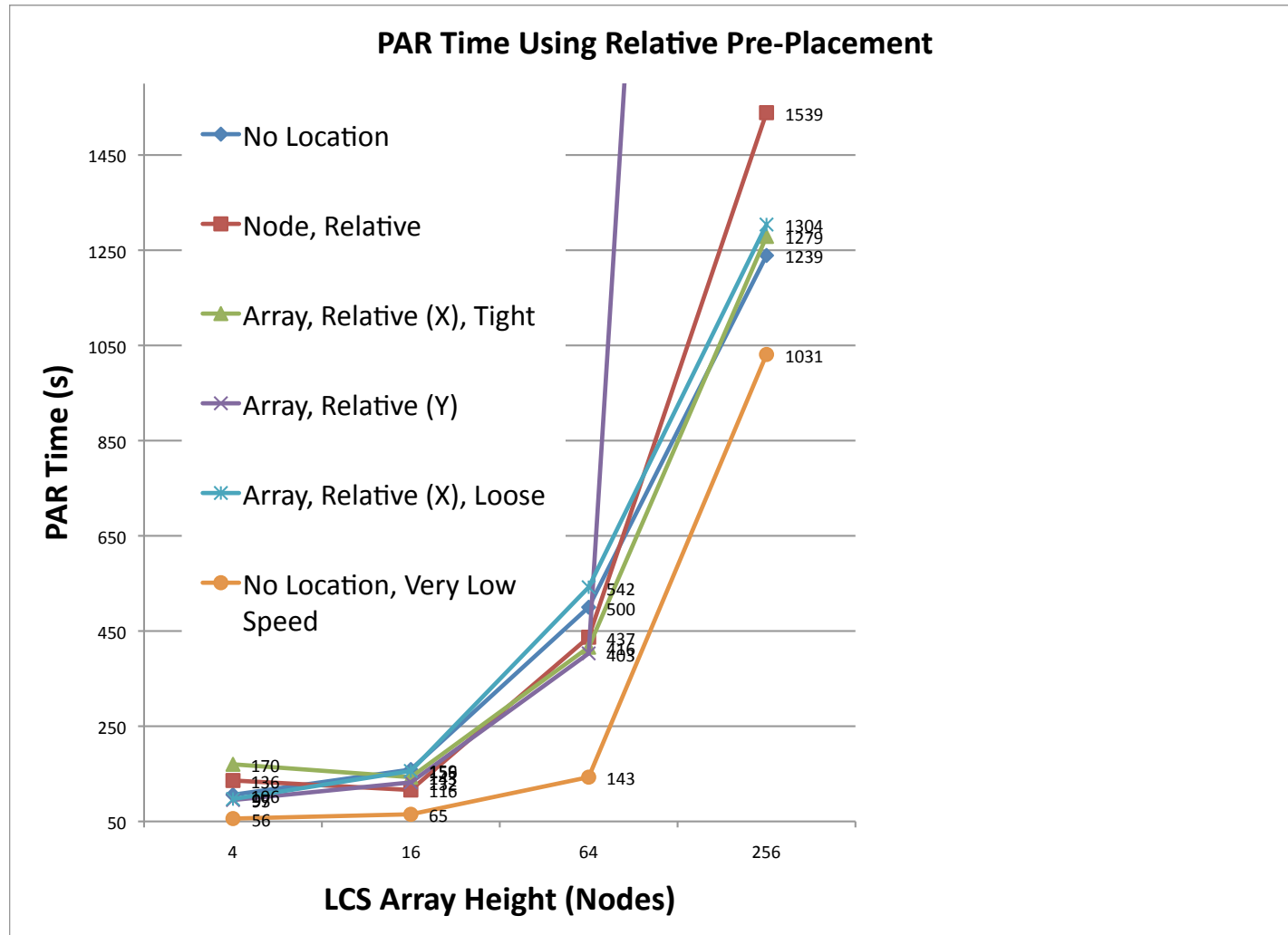


2. Fast Compile / Debug

- Slow P&R is a major hindrance to application development - not just an annoyance (FPGAs have subtle tradeoffs between size, performance, and energy efficiency).
- What can be done about it? Can P&R be minutes or seconds instead of days or hours?

2. Fast Compile / Debug

- Design(er) driven placement:
 - “Placement is the slow part”



2. Fast Compile / Debug

- Parallelization of Place/Route:

A. Choong et al., “Parallelizing simulated annealing-based placement using GPGPU,” in FPL, Aug. 2010, pp. 31–34.

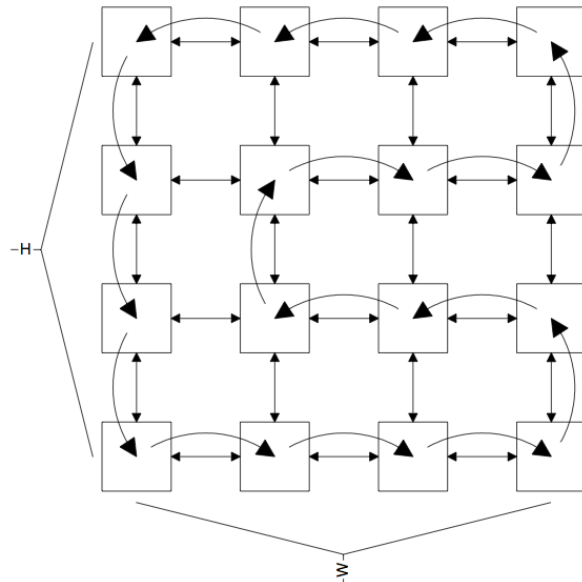
*“Scalable and Deterministic Timing-Driven Parallel Placement for FPGAs”, Chris Wang, Guy Lemieux, FPGA2011 **12x speedup with 25 thread***

Parallelization of Routing possible also.

Off course, not mainstream yet.

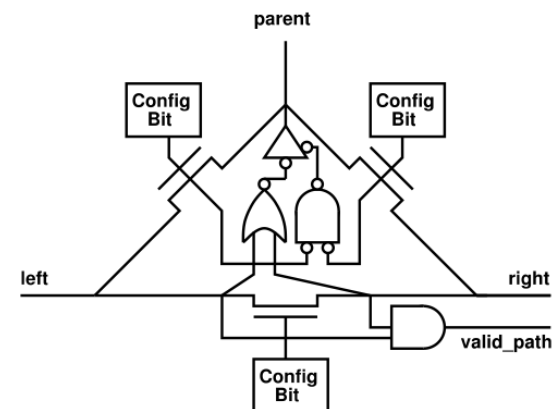
2. Fast Compile / Debug

- Hardware Assisted Place/Route:



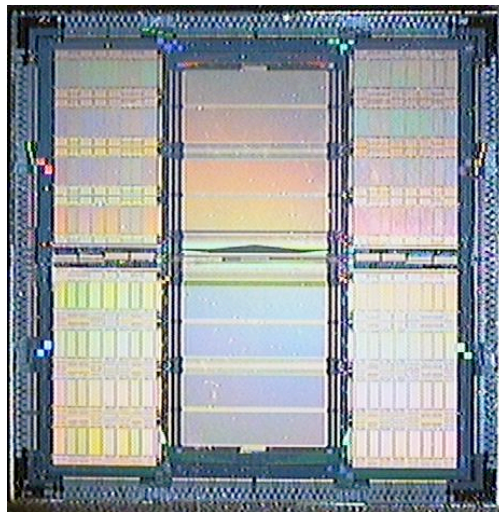
*M. G. Wrighton and A. M. DeHon,
“Hardware-assisted
simulated annealing with
application for fast FPGA
placement,” in FPGA, 2003.
~1000X speedup (ms range)*

*A. DeHon, R. Huang, and J.
Wawrzynek. “Stochastic Spatial
Routing for Reconfigurable Networks,”
Journal of Microprocessors and
Microsystems, Volume 30, Issue 6, 4
Sep 2006, pp. 301-318. (ms range)*

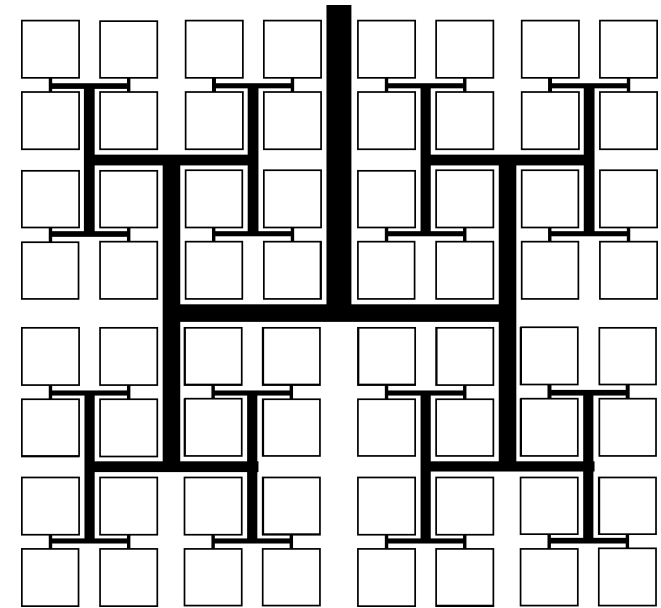


2. Fast Compile / Debug

- Change the Architecture
 - Hierarchical architecture can help P&R Times
 - Placement become a successive partitioning : much faster P&R times



HP Teramac FPGA Chip



3. Standard “seamless” operating environment

- Xilinx EDK, Altera Nios-II EDS (Still hard to use)
- Convey, BEEcube (must buy into their model)
- A few academic FPGA OS efforts:

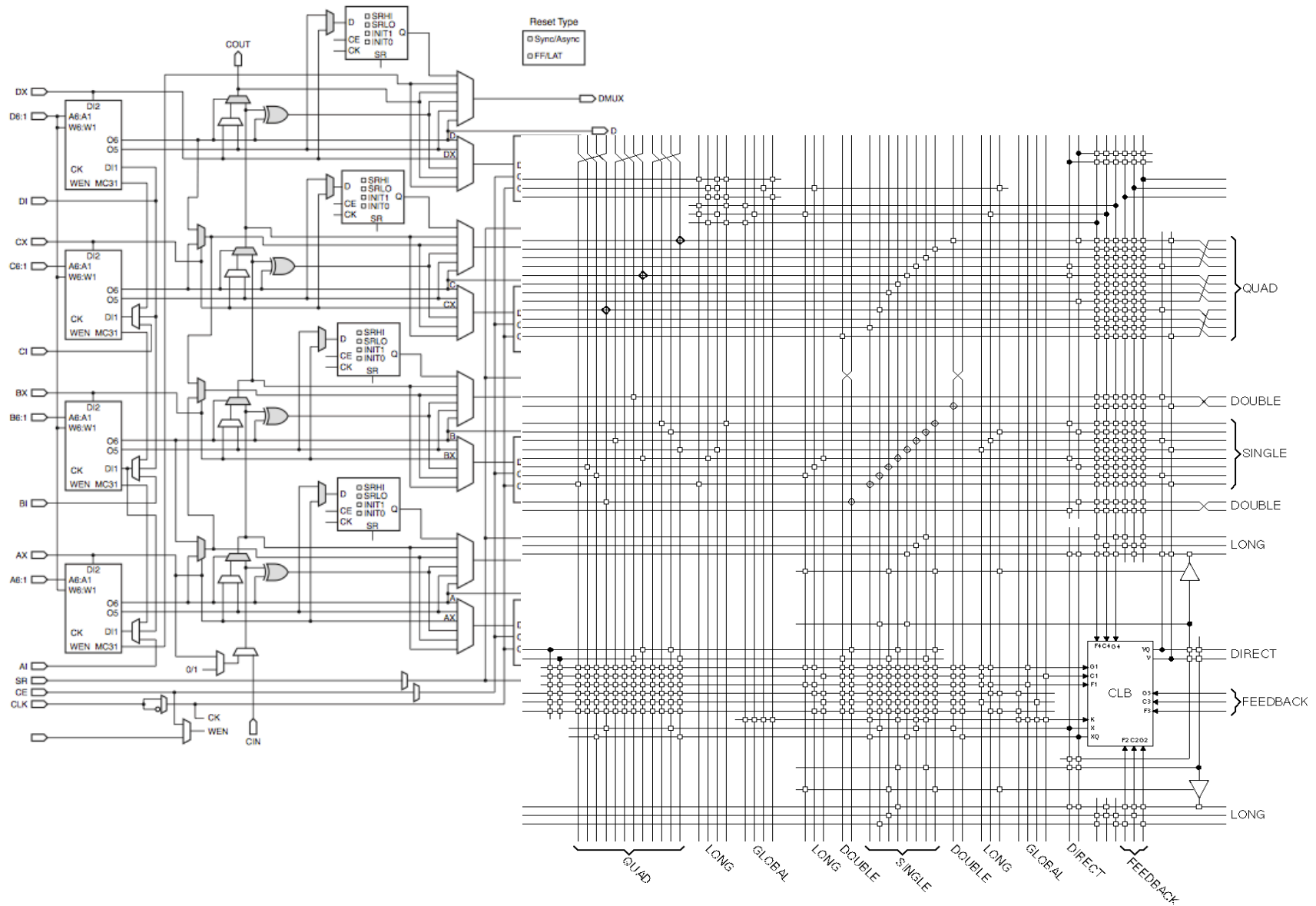
*G. B. Wigley, D. A. Kearney, and D. Warren, “Introducing **ReConfigME**: An operating system for reconfigurable computing,” in Proceedings of the 12th International Conference on Field Programmable Logic and Application (FPL’02). Springer, 2002.*

BORPH: An Operating System for FPGA-Based Reconfigurable Computers

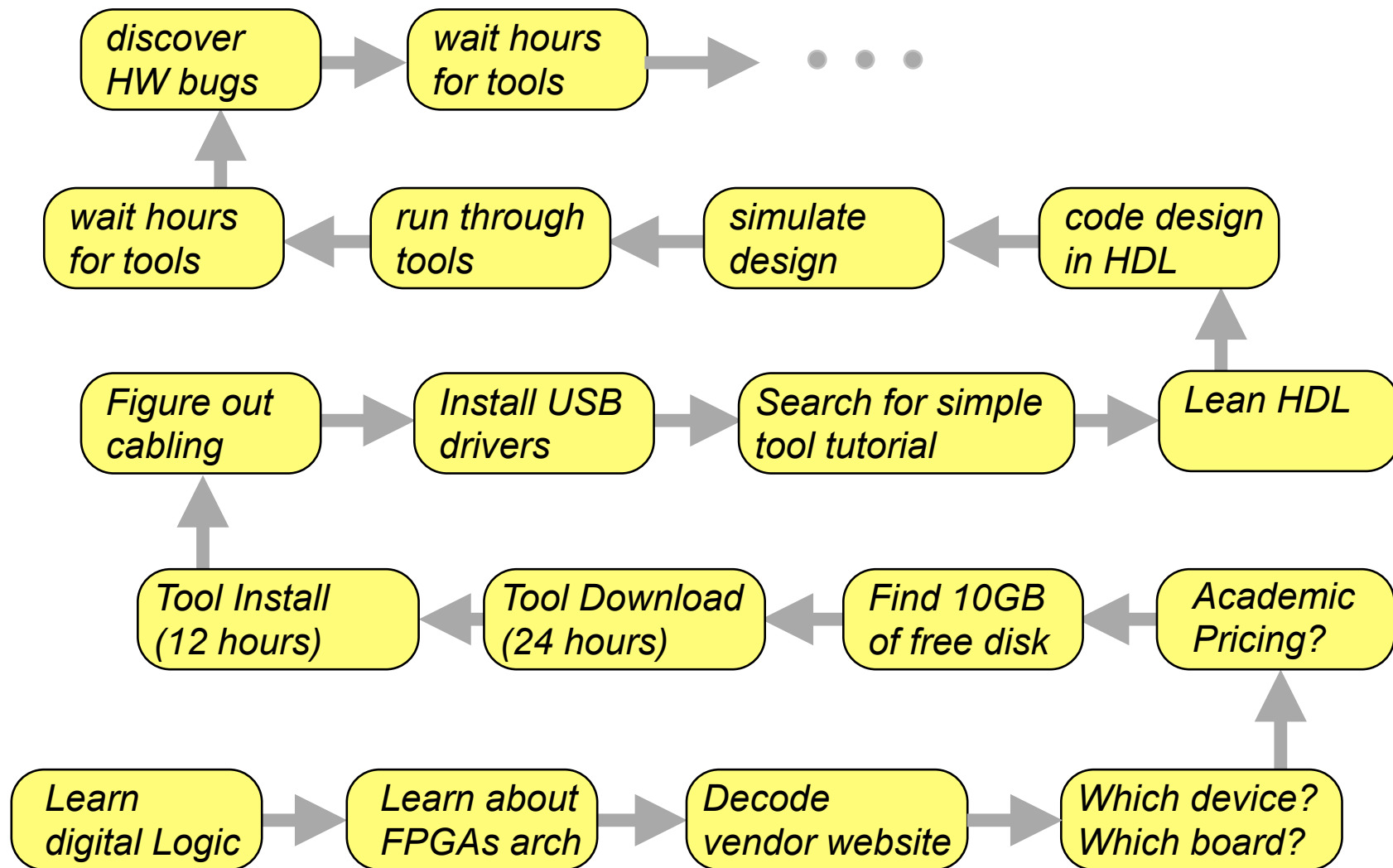
Hayden Kwok-Hay So and Robert W. Brodersen, Berkeley

LEAP: A Virtual Platform Architecture for FPGAs, Angshuman Parashar (Intel); Michael Adler (Intel); Kermin Fleming (MIT); Michael Pellauer (MIT); Joel Emer (Intel/MIT)

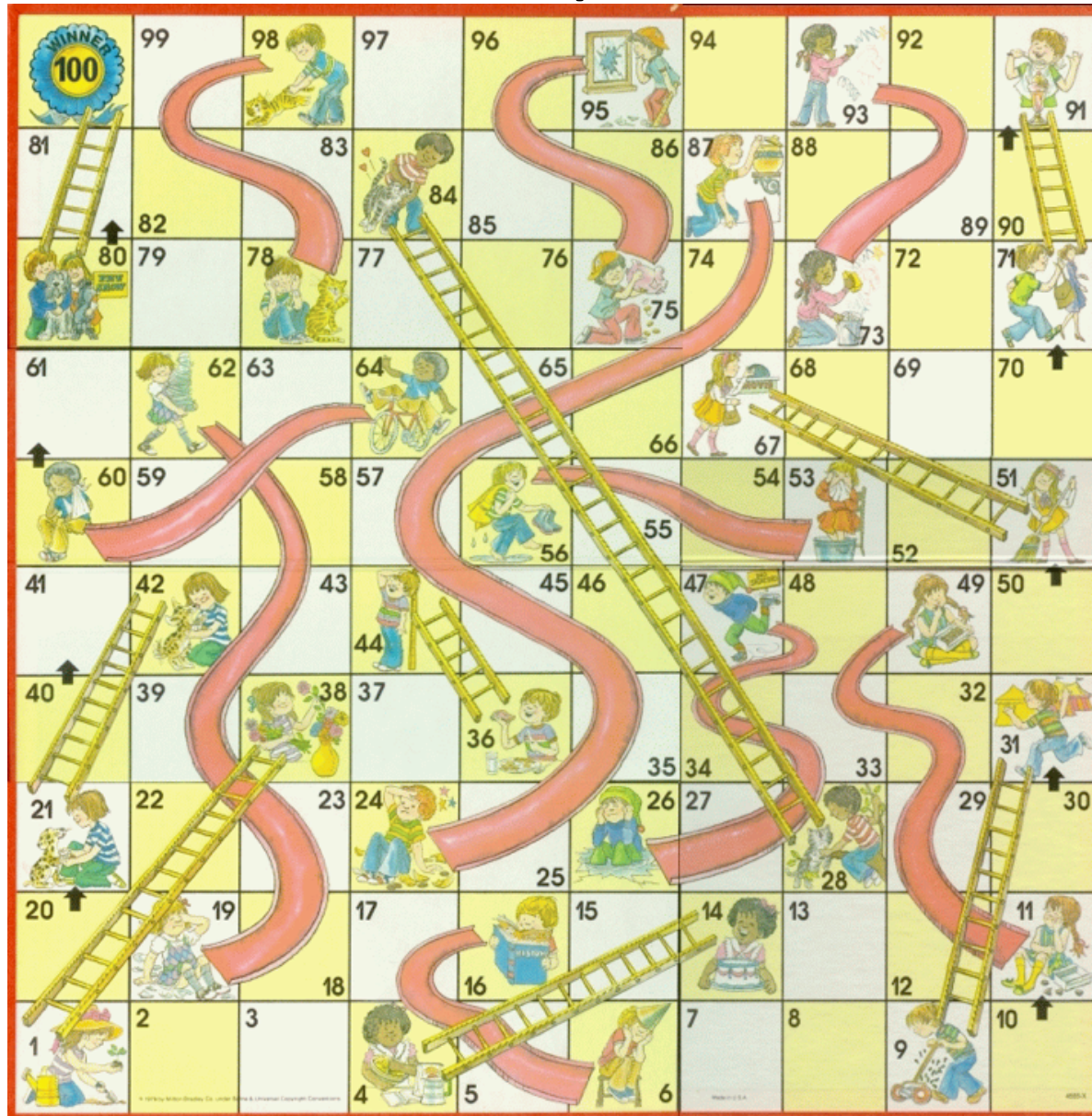
Current FPGAs very complex!



So you want to compute with FPGAs?



So you want to compute with FPGAs?



“Ease of Use” Report Card

	Micro-processor	GPGPU	FPGA
Programming Models	A	B	C
Fast Compile/Debug	A	B	F
Operating Environment	A	A	F

Why are FPGAs struggling as computing devices?

- No standardization of hardware abstraction (no “ISA”) –nor a standard hardware platform
- No open documented binary interface
- Very complex hardware (and Hardware model).

What we Need.

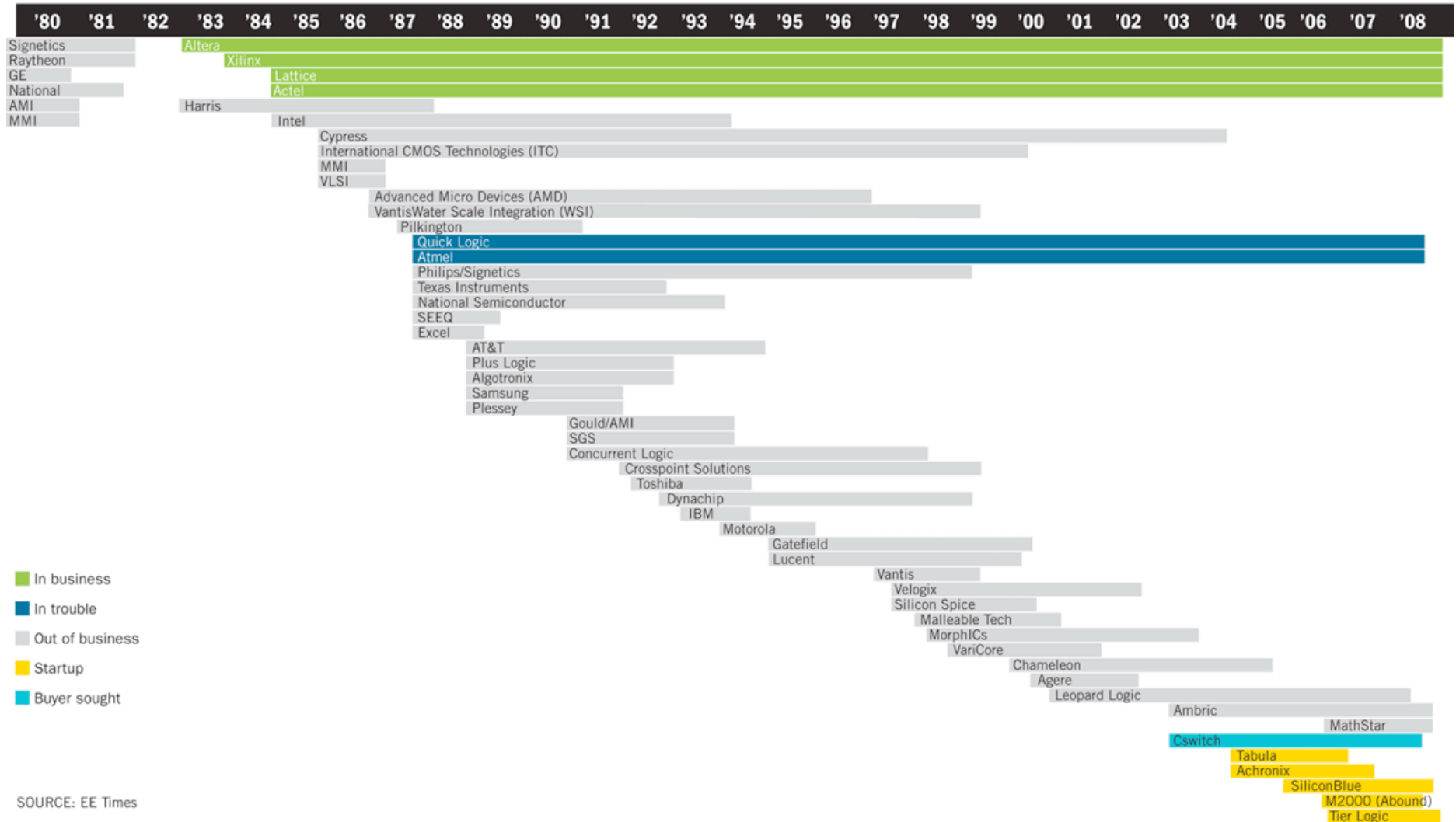
- A simple, open, easy to understand, easy to use FPGA architecture with features for computing.
- Supported by open Source Tools.
- (If ISA style abstraction doesn't work for FPGAs, then settle for simple array design.)

“Perhaps Xilinx/Altera can Help”

- Probably not.
- Focus on ASIC replacement NOT computing.
- Open binary interface not in their interest:
 - must protect software investment
 - must control architecture evolution
- Monotonically increasing complexity because of competition

“Perhaps a Startup will Help”

History of PLD startups

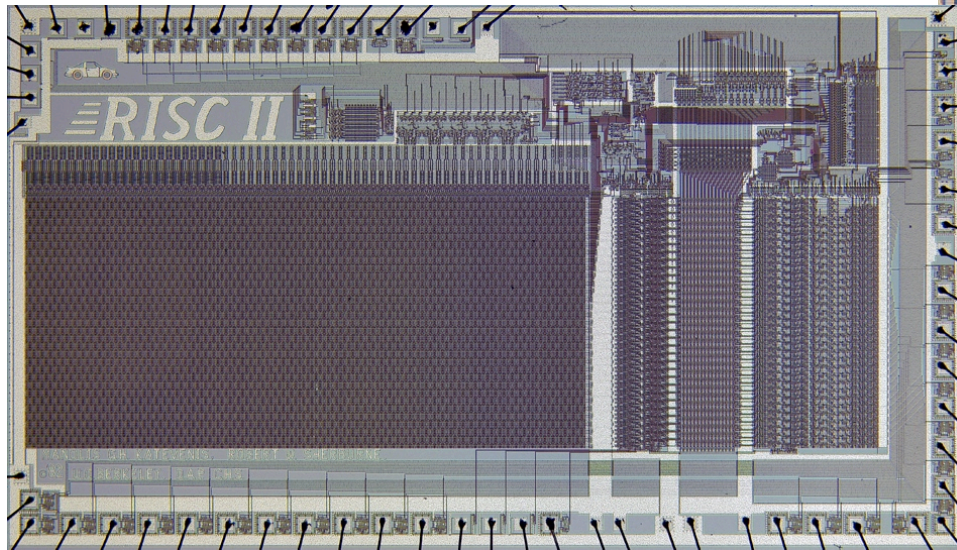
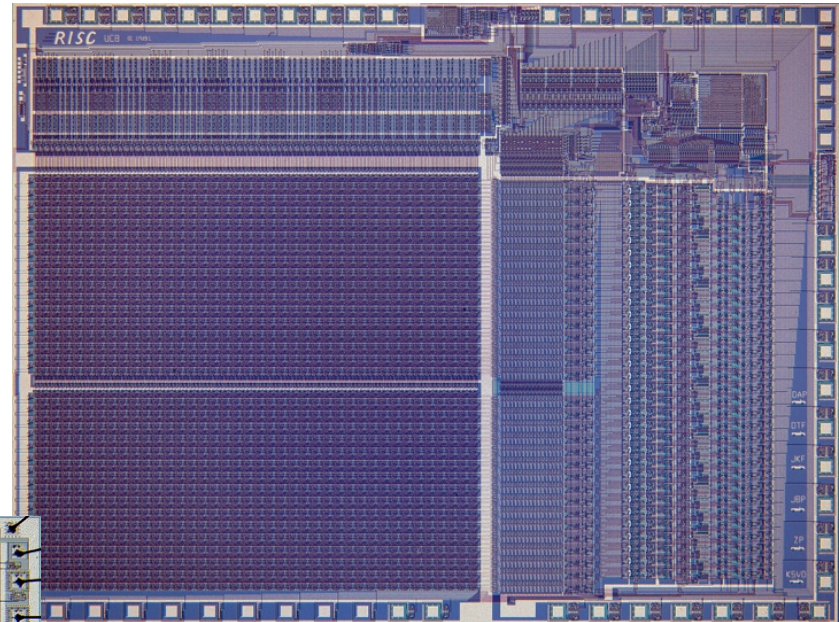


Historical Analogy, early 1980's

- Processor ISAs property of companies. Dominated by a few: IBM, DEC, Intel,
- Compilers written and owned by companies. Closed source.
- High performance applications used low-level programming techniques (assembly language programming).
- MicroProcessors had evolved from simple (4004) to be very complex (VAX example: variable length instructions, dozens of memory addressing modes, VERY complex opcodes such as polynomial evaluation!).
- Hardware designers drove the processor evolution process rather than application developers.
- *Nearly impossible for Universities to participate in meaningful computer architecture research - impossible to build a chip at the complexity level of a VAX or even x86.*

Berkeley RISC / Stanford MIPS

- RISC-I (1982) Contains 44,420 transistors, in 5 micron NMOS, with a die area of 77 mm², ran at 1 MHz. This chip is probably the first VLSI RISC.



RISC-II (1983) contains 40,760 transistors, was fabbed in 3 micron NMOS, ran at 3 MHz, and the size is 60 mm².

Innovation Started Happening in the Universities

- Simplified architecture possible to implement and modify
 - universities now can participate in architecture research
- Inspired compiler work since optimizing compiler now within reach, led to open tools ..
- Academic community results filter back to industry

- SPARC at sun
- MIPS company



- MIPS still widely used in Academia.

Time to Reset!



"Call to Arms : Volunteer Wanted"

*University Consortium to develop
open simple FPGA HW and SW*

- Simple architecture is good enough.
 - Simpler tools, run faster
 - Easier to understand
 - Easier to migrate to new process
 - (Silicon Blue, Achronix both have a simple "retro" architecture.)
- First step is to write the classic ISA analysis paper. Figure out how to simplify.
- Eventually could create a new startup model, in style of Redhat.

Thanks!

Contact: johnw@eecs.berkeley.edu