Adventures with a Reconfigurable Research Platform

John Wawrzynek
Computer Science Division &
Berkeley Wireless Research Center
UC Berkeley
BEE2 compute module

HW Completed Dec. 2004 (1 year from concept)
Supporting “gateware” another 1.5 years!
Application development continues to this day.
A Sea Change in Processor Design

- **Sea change in chip design: multiple “cores”**
  - More instances of simpler processors are more power efficient
  - Simple model for hardware scaling
- **Moore’s Law and many-cores: 2X CPUs per chip / ~ 2 years**
- **Multicores now (2, 4, 8, 16), Manycore soon (64, …)**

---

*Uniprocessor SpecInt Performance:*

- Power Wall
  - +
  - Memory Wall
  - +
  - ILP
  - =
  - Brick Wall

Only 2x / 5yrs
Teraflops Research Chip
100 Million Transistors • 80 Tiles • 275mm²

First tera-scale programmable silicon:

– Teraflops performance
– Tile design approach
– On-die mesh network
– Novel clocking
– Power-aware capability
– Supports 3D-memory

Not designed for IA or product
Is the processor the new “transistor”?

- Intel 4004 (1971): 4-bit, 2312 transistors, 0.4 MHz, 10 micron PMOS, 11 mm²

- Berkeley RISC II (1983): 32-bit, 5 stage pipeline, 40,760 transistors, 3 MHz, 3 micron NMOS, 60 mm² chip

- 125 mm² chip, 0.065 micron CMOS = 2312 RISC II+FPU+Icache+Dcache
  - RISC II shrinks to ~ 0.02 mm² at 65 nm
Or is the Processor the new “LUT”?

- **First Xilinx FPGA (1985), XC2064 had 64 4-LUTs!**
- **Tilera TILE64 device, announced last week has 64 VLIW processor cores**
Problems with “Manycore” Sea Change

Algorithms, Programming Languages, Compilers, Operating Systems, Architectures, Libraries, … not ready for 100-1000 CPUs / chip

Software people don’t start working hard until hardware arrives

3 months after HW arrives, SW people list everything that must be fixed, then we all wait 4 years for next iteration of HW/SW

How do we innovate in timely fashion on 1000 CPU systems in algorithms, compilers, OS, architectures, … without waiting years between HW generations?
Vision: Build MPP from FPGAs

- As ~ 10 CPUs will fit in a Field Programmable Gate Array (FPGA), 1000-CPU system from ~ 100 FPGAs?
  - FPGA generations every 1.5 yrs; ~2X CPUs, ~1.2X clock rate
- Architecture research community does logic design of “gateware” to create out-of-the-box, MPP that runs standard binaries of OS and applications
  - Gateware: Processors, Caches, Coherency, Switches, Routers, Ethernet Interfaces, ...
  - Goal: 1000 processor, standard ISA binary-compatible, 64-bit, cache-coherent MPP @ 200 MHz/CPU in 2008
- 3rd party sells RAMP hardware at low cost
- Reproduce success of SimpleScalar simulation platform + provide early parallel platform for software.

RAMP “Research Accelerator for Multiple Processors”
Multiprocessing Watering Hole

- RAMP attracts many communities to shared artifact
  ⇒ Cross-disciplinary interactions
  ⇒ Ramp up innovation in multiprocessing
- RAMP as next Standard Research Platform?
  (e.g., VAX/BSD Unix in 1980s)

Parallel file system  Dataflow language/computer  Data center in a box
Thread scheduling  Security enhancements  Internet in a box
Multiprocessor switch design  Router design  Compile to FPGA
Fault insertion to check dependability  Parallel languages
RAMP Provides High Visibility for FPGAs

- Today, FPGAs get used in undergraduate hardware courses and by a few graduate research efforts.
- RAMP exposes FPGAs to many more grad students and faculty research programs
  - Parallel Compilers and Languages
  - Operating systems and File Systems
  - Computer Architecture
  - Distributed Computing/Networking (“Internet-in-a-box”)
  - All see reconfigurable logic as an emulation platform, and opens up possibility for innovation using FPGAs.

Good for FPGA vendors and the Field Programmable Logic and Reconfigurable Computing Community - good for advancing the state-of-art in reconfigurable hardware and software.
Collaborators

- Initial RAMP Participants/Developers:
  Krste Asanović (MIT), Derek Chiou (UT Austin), James Hoe (CMU),
  Christos Kozyrakis (Stanford), Shih-Lien Lu (Intel), Mark Oskin
  (Washington), David Patterson (Berkeley), Jan Rabaey (Berkeley),
  and John Wawrzynek (Berkeley) PI

- RAMP hardware development activity centered at BWRC.

- NSF grant for staff.

- Major commitment from Xilinx

- Integral part of FCRP/GSRC

- Collaboration with MSR (Chuck Thacker) on RAMP2 FPGA platform.

- Sun, IBM actively contributing (cores, student support)
### Why RAMP Good for Research MPP?

<table>
<thead>
<tr>
<th></th>
<th>SMP</th>
<th>Cluster</th>
<th>Custom</th>
<th>Simulate</th>
<th>RAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalability (1k)</td>
<td>C</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Cost (1k CPUs)</td>
<td>F (20M)</td>
<td>C (1M)</td>
<td>F (3M)</td>
<td>A+ (-0M)</td>
<td>A (-0.1M)</td>
</tr>
<tr>
<td>Cost to own</td>
<td>A</td>
<td>D</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Power/Space (kw, racks)</td>
<td>D (120 kw, 6 racks)</td>
<td>D (120 kw, 6 racks)</td>
<td>A (100 kw, 3 racks)</td>
<td>A+ (.1 kw, 0.1 racks)</td>
<td>A (1.5 kw, 0.3 racks)</td>
</tr>
<tr>
<td>Community</td>
<td>D</td>
<td>A</td>
<td>F</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Observability</td>
<td>D</td>
<td>C</td>
<td>D</td>
<td>A+</td>
<td>A+</td>
</tr>
<tr>
<td>Reproducibility</td>
<td>B</td>
<td>D</td>
<td>B</td>
<td>A+</td>
<td>A+</td>
</tr>
<tr>
<td>Reconfigurable</td>
<td>D</td>
<td>C</td>
<td>D</td>
<td>A+</td>
<td>A+</td>
</tr>
<tr>
<td>Credibility</td>
<td>A+</td>
<td>A+</td>
<td>A-</td>
<td>F</td>
<td>B</td>
</tr>
<tr>
<td>Perform. (clock)</td>
<td>A (2 GHz)</td>
<td>A (3 GHz)</td>
<td>B (.4 GHz)</td>
<td>F (0 GHz)</td>
<td>C (.1 GHz)</td>
</tr>
<tr>
<td>GPA</td>
<td>C</td>
<td>B-</td>
<td>C+</td>
<td>B</td>
<td>A-</td>
</tr>
</tbody>
</table>
RAMP 1 Hardware
Chen Chang, Pierre-Yves Doz

- 14x17 inch 22-layer PCB
- 5 Virtex II FPGAs, 20 banks DDR2-400 memory, 18 10Gbps links
- Administration/maintenance ports:
  - 10/100 Enet
  - HDMI/DVI
  - USB
- ~$6K (w/o FPGAs or DRAM or enclosure)

BEE2: Berkeley Emulation Engine 2
BEE2 Module Details

- Four independent 200MHz (400 DDR2) SDRAM interfaces per FPGA
- FPGA to FPGA connects using LVCMOS
  - Designed to run at 300MHz
  - Tested and routinely used at 200MHz (SDR) - single ended

- “Infiniband” 10Gb/s links use “XAUI” (IEEE 802.3ae 10GbE specification) communications core for the physical layer interface.
- Hardware will support others, ex: Xilinx “Aurora” standard, or ad hoc interfaces.
RAMP Development Plan

1. Distribute systems internally for RAMP 1 development
   - Xilinx paying for production of a set of modules for initial contributing developers and first full RAMP systems

2. Build & release publicly available out-of-the-box MPP emulator reference designs
   - Based on standard ISA (Sun SPARC, IBM Power, …) for binary compatibility
   - Complete OS/libraries
   - Others locally modify RAMP as desired

3. Design next generation platform for RAMP 2
   - Base on 65nm FPGAs (2 generations later than Virtex-II)
   - 3rd party to build and distribute systems (at low-cost), open source RAMP gateware and software
   - Hope RAMP 3, 4, … self-sustaining
Which ISA to pick?

- Long-term goal is replaceable ISA/CPU L1 cache, rest infrastructure unchanged (L2 cache, router, memory controller, …)
- What do you need from a CPU?
  - Standard ISA (binaries, libraries, …), simple (area), 64-bit (coherency), DP Fl. Pt. (apps), verification suites, efficient FPGA implementation
- ISAs so far (Synthesizable HDL)
  - Got it: Power 405 (32b), SPARC v8 (32b), SPARC v9 (Niagara, 64b), SPARC Leon, Xilinx Microblaze (32b)
  - Likely: IBM Power 64b, ARM
  - Probably (haven’t asked): MIPS32, MIPS64
  - Unlikely: x86
  - Even less likely: x86-64
### Which soft-core to pick?

<table>
<thead>
<tr>
<th></th>
<th>IBM</th>
<th>Freescale</th>
<th>Leon3</th>
<th>OpenSPARC T1 (Niagara)</th>
<th>ProtoFlex</th>
<th>Simply RISC S1</th>
<th>QEMU/FAST functional</th>
<th>MicroBlaze (v5.0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISA</td>
<td>PPC405</td>
<td>PPC</td>
<td>SPARC V8</td>
<td>UltraSPARC Arch 2005 (SPARC V9+)</td>
<td>SPARC V9</td>
<td>UltraSPARC Arch 2005 (SPARC V9+)</td>
<td>x86</td>
<td>MIPS-like RISC (w/o MMU)</td>
</tr>
<tr>
<td>FPU</td>
<td>no</td>
<td>no (available)</td>
<td>Yes (no source code)</td>
<td>Yes</td>
<td>not yet</td>
<td>Yes</td>
<td>software</td>
<td>partial IEEE 754 single precision</td>
</tr>
<tr>
<td>microarchitecture</td>
<td>5-stage, in-order, single-issue</td>
<td>7-stage, in-order?</td>
<td>7-stage, in-order pipeline</td>
<td>8-core, 32-thread, 6-stage pipe (1 core/1 thread in initial FPGA version)</td>
<td>ISA only</td>
<td>1-core, 4-thread derivative of OpenSPARC T1</td>
<td>ISA</td>
<td>3-stage in-order</td>
</tr>
<tr>
<td>licensing</td>
<td>restricted</td>
<td>yes</td>
<td>LGPL</td>
<td>GPL</td>
<td>SIMICS</td>
<td>GPL</td>
<td>LGPL</td>
<td>Xilinx EDK</td>
</tr>
<tr>
<td>speed</td>
<td>30MHz</td>
<td>25MHz</td>
<td>50MHz (0.85 MIPS/MHz)</td>
<td>25 MHz (Virtex-II)</td>
<td>&lt;10MIPS</td>
<td><em>?</em></td>
<td>5MIPS</td>
<td>&gt;100 MHz (0.5 MIPS/MHz)</td>
</tr>
<tr>
<td>size, in 4-input LUTs (w/o FPU)</td>
<td>20K</td>
<td>33.5K</td>
<td>8K</td>
<td>135K (full 8-Core), 48K (1 thread/1 core, placed/routed)</td>
<td>13K (partial core)</td>
<td><em>?</em></td>
<td>PowerPC</td>
<td>2.8K</td>
</tr>
<tr>
<td>HDL</td>
<td>Verilog</td>
<td>Verilog</td>
<td>VHDL</td>
<td>Verilog</td>
<td>Bluespec</td>
<td>Verilog</td>
<td>Verilog</td>
<td>Encrypted VHDL</td>
</tr>
<tr>
<td>verification suite</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>Niagara (OpenSPARC)</td>
<td>Niagara</td>
<td>Niagara (OpenSPARC)</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

Long-term goal is replaceable CPU L1 cache, rest unchanged (L2 cache, router, mem. controller, ...)

8/28/2007

Wawrzynek, FPL 2007
RAMP Gateware Design Framework

Problems:

1. **Hardware Scripting**
   - Components developed all over the country in a variety of HDLs, some might even be in software.

2. Cycle-accurate distributed emulation with **time dilation**
   - FPGA *host* might need multiple cycles for a single cycle of emulated *target* component: to save resources, experiment with ratio of network/memory/processor speeds.
   - Distributed components must stay synchronized.

3. Hide hardware platform details (physical links, memory interface).
   - Portability, productivity, ...

**An Insight:** most large building block fits inside single FPGA today. What doesn’t, is between chips in real design ⇒ no need to partition across FPGAs
RAMP Description Language (RDL)
Greg Gibeling, Andrew Schultz, Krste Asanovic

- Design composed of *units* that send messages over channels via ports
- Units (> 10,000 gates)
  - CPU + L1 cache, DRAM controller, ...
- Channels (~ FIFO)
  - Lossless, point-to-point, unidirectional, in-order message delivery...

1. Units in any hardware design languages (will work with Verilog, VHDL, BlueSpec, C, ...)
2. RDL hides details of communication (automatically maps to appropriate physical link)
3. Carefully accounts for *Target Clock Cycles*, effectively time-stamps messages.

*Similar to other “process network” models (KPN) with explicit management of “target” clock cycles.*
RAMP Description Language (RDL)

- Monitoring
  - All communication is over channels
    - Can be examined and controlled
  - Target time can be paused or slowed

- Injection
  - Makes developing test benches easy
    - Simply inject a sequence of messages

- Upcoming
  - Activity monitors for power estimation.

```
Monitoring
- All communication is over channels
  - Can be examined and controlled
- Target time can be paused or slowed

Injection
- Makes developing test benches easy
  - Simply inject a sequence of messages

Upcoming
- Activity monitors for power estimation.
```
RAMP Prototypes

- RAMP design driver projects:
  1) **RAMP Red**: Port of existing transactional cache system to FPGA PowerPC cores
  2) **RAMP Blue**: Message passing distributed memory system using an existing FPGA optimized soft core
  3) **RAMP White**: Cache coherent multiprocessor system with full featured soft-core

- Goal:
  - Reuseable gateware.
  - Starting points for others outside of initial group of collaborators.
First MP with HW support for transactional memory

- Embedded PowerPC 405 hardcores, connecting to a shared-memory over a packet-switched network.
- Custom cache (emulated) has transactional memory support.
- Linux runs on one core and 8 used for applications, proxy kernels forward exceptions and syscalls.
- Programming model: multithreaded C or Java with locks replaced by transactional constructs.
- Includes extensive HW/SW debugging/monitoring framework.

100MHz on BEE2:
100X faster than software simulation on 2GHz workstation.
RAMP Blue
Andrew Schultz, Alex Krasnov

**V1.0:**
- Used 8 BEE2 modules
- 4 user FPGAs of each module held 100MHz Xilinx MicroBlaze soft cores running uCLinux.
- 8 cores per FPGA, **256 cores total**
- Dec 06: 256 cores running benchmark suite of UPC NAS Parallel Benchmarks

**V3.0:**
- Uses 21 BEE2 modules
- 12 cores per FPGA, **1008 cores total**
- Cores running at 90 MHz
- FPGA Mesh topology
- Scaled version demo today

**Future versions**
- Use newer BEE3 FPGA platform
  Support for other processor cores.
RAMP Blue Demonstration

- NASA Advanced Supercomputing (NAS) Parallel Benchmarks (all class S).
- UPC versions (C plus shared-memory abstraction)
  - CG Conjugate Gradient
  - EP Embarassingly Parallel
  - IS Integer Sort
  - MG Multi Grid

Virtual network traffic
Physical network traffic
8/28/2007
Wawrzynek, FPL 2007
Distributed Shared-Memory Machine.

- Components from Red and Blue with others (integration at UT Austin)
- Version 1 simple ring network with directory-based coherence engine.
- Each processor connects to an Intersection Unit (IU) for access to a memory controller (MCU), a network interface unit (NIU), and an optional I/O.

- Development of a simple embedded PowerPC core version with snoopy cache (from Red) underway. Later Leon Sparc core and directories-based coherence, more interesting network topologies.
RAMP and Reconfigurable Computing

- RAMP is not a “reconfigurable computing” project, per se. Applications are not directly mapped to FPGAs.
- Focus is on building a research platform (with observability, controllability, etc.) not application performance results.
- In RAMP FPGAs are used to advance a technology, that ironically might compete.

“I’m not interested in computing on multi-core architectures, I want to compute on FPGAs.”

Ivo Bolsens
Xilinx CTO
Manycore as a “programming model”

- Reconfigurable computing has suffered from lack intermediate representations:
  - And thus programming models and related tool support.
  - The gap between application and HW is too large.
- Array of processor cores is an “established model” with available compilers, libraries, etc.
- Therefore, manycore might be a “stepping stone” to help in the reconfigurable computing vision.
- Applications start as software only on soft processors and over time pieces migrate to gateware.
Arrays of Soft Processors on FPGAs

_Increasingly popular idea; More and more conference papers each year._

**Advantages**
- Mature programming and architectures (at least in domain of scientific computation)
- Ride the wave of interest, tools, libraries, application mapping on multi- and manycore.

**Disadvantages**
- Density/Speed: 10X in area, 10X in frequency
- Parallel programming is no picnic (still largely unsolved!) Similar to circuit design.

_Through gateware acceleration, can we compensate for the 100x decrease in area X frequency? If not, we’re in trouble!_
“Attack of the Killer Micros”

Eugene Brooks, 1989

- In the 80’s there was a huge push behind microprocessors. They nearly crushed anything in their path - custom processors for supercomputing, custom VLSI architectures, ...

- Now there is a huge push behind manycore technology and business.

- Same arguments that we have used for reconfigurable computing is being used by manycore.
Call to Arms for Community

Manycore and other spatial arrays are a real thread to the survival of reconfigurable computing.

- Must understand and demonstrate the relative strengths of reconfigurable fabrics and show how they complement (or replace in some cases) manycore architectures.
- The advantages need to go beyond spatial computation, as many competing arrays are emerging.
- Many models possible here:
  - Processor core/accelerator hybrids
  - RC network fabrics, and memory hierarchies
  - … your idea here …
- We are looking at a (rare) few year window for architecture innovation.
Current BEE2 Users (partial list)

- **UC SSL, Radio Astronomy Lab**: SETI, Allen Telescope Array
- **RAMP**: UCB, Stanford, UW, UT Austin, CMU, MIT, Intel: Multiprocessor Emulation
- **BWRC**: ASIC/SOC emulation, Cognitive Radio Algorithm Exploration, LDPC simulation, Media processing
- Paul Chow / U Toronto: Molecular Dynamics
- Bob Conn/ Research Triangle Inst.: Spice Circuit Simulation
- Rob Reutenbar/CMU: Speech Recognition
- Wing Wong / Stanford BioInformatics Group: Biological signaling research
- Chris Dick, Kees Vissers / Xilinx: Signal/Media Processing
- **DARPA/DOD** Contractors
- Wen-mei Hwu / UIUC, IMPACT Multicore Prototyping
- K. Constantinides, T. Austin, V. Bertacco, S. Mahlke / U Michigan, Reliability Analysis Framework
- David August, Sharad Malik, Margaret Martonosi, and Li-Shiuan Peh / Princeton. Concurrent Architectures and Runtime Validation through Hardware Emulation
- Tim Cheng / UC Santa Barbara, Fault Emulation
- Josie Ammer / University of Washington, Emulation of an 802.11a Physical Layer for Development and Evaluation of Robust DSP Functional Units
- Mary Jane Irwin / Pennsylvania State University, Multi-Core Architectures
- M. Horowitz and C. Kozyrakis / Stanford EECS, Silicon Compiler for Domain Specific Systems
BEE2 Sites: August 2007

62 BEE2 modules to date. ~20 more currently in production. More on the way to Europe, South America, and Asia.
Radio Astronomy

Signal Processing, Dataflow oriented, limited precision
Low volume (NRE dominated), Flexibility a must.

Arecibo Observatory
Large-N, Small-D Telescope Concept

- Use lots of small diameter antennas to achieve large aggregate collecting area
  - Flexible usage model, multi-user, multi-subarrays
  - Reliability through redundancy
  - Lower overall system cost.

![Graph showing cost versus antenna diameter]

- Cost ($)
- Compute Hardware Cost Dominate
- Antenna Steel Cost Dominate
- Moore’s Law
- Antenna diameter
Signal Processing for Allen Telescope Array
Dan Werthimer, SSL & Berkeley CASPER Group

Image-formation ($n^2$ cross-correlation / iFFT), SETI spectroscopy, etc.
Peta-Ops desired.

ATA at Hat Creek Observatory
SETI Billion Channel Spectrometer
Chen Chang

- 1 billion Channel real-time spectrometer ➔ 0.7Hz channels over 800MHz
- Implemented on one BEE2 module and yields 333GOPS (16-bit mults, 32-bit adds), at 150Watts
Real-time Correlation & Image Formation

Development of a general architecture for ATA correlation and image formation is underway.

*Leverages commercial GigE switches.*

<table>
<thead>
<tr>
<th>Antenna array size</th>
<th>32</th>
<th>206</th>
<th>350</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baselines (# correlations)</td>
<td>496</td>
<td>21115</td>
<td>61075</td>
</tr>
<tr>
<td>BEE2 modules (PFB)</td>
<td>1</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>BEE2 modules (XMAC)</td>
<td>1</td>
<td>43</td>
<td>123</td>
</tr>
<tr>
<td>BEE2 modules (Imager)</td>
<td>1</td>
<td>29</td>
<td>82</td>
</tr>
<tr>
<td>Digitizers</td>
<td>4</td>
<td>26</td>
<td>44</td>
</tr>
<tr>
<td>Ethernet switches</td>
<td>2</td>
<td>31</td>
<td>86</td>
</tr>
</tbody>
</table>

Notes: 100MHz band from each antenna

Dual polarization

512x512 pixels image

1024 frequency channels per pixel
Dealing with “Accidental Success”

- “How do I buy one?”
- How does the university sell these?
- No “Design for Manufacturing” or volume production:
  - Tight spacings
  - Thick board, DIMM socket problems
  - No standard power supply
  - No chassis
- How do we support it?
Berkeley BEE2 Staff

Dan Burke
BEE2 Hardware
HW/SW Interface
Linux OS
drburke@berkeley.edu

Brian Richards
Design Tools
Matlab/Simulink
richards@eecs.berkeley.edu

Others: Henry Chen (hchen05@ssl.berkeley.edu), Chen Chang (chenzh@EECS.Berkeley.EDU), BEE2-support@eecs.berkeley.edu
Support

- Documentation
- HW Test-suites
- Common Gateware
  - Memory controllers
  - Chip-to-chip communication blocks
- Design Tools
News

May 16, 2007 ~ This is the new home for all FPGA related open source design repository research uses. All users have read access to all the designs in the repository, and registered repository as well. Please check back this page often for latest improvements and newly join.

Repository Usage

CVS Access

For read-only CVS access, please follow the CVS link button above for a ViewVC web interface of the repository, including dynamic tar.gz capability. For CVS write or read/write access, contact repository-admin@lists.berkeley.edu and explain the research project involved. CVS login requires proper login accounts.

Repository Documentation

http://repository.eecs.berkeley.edu

The ViewVC web interface has been modified to provide a built-in documentation mechanism to allow a simple documentation page to be attached to any directory level in the CVS tree. Please follow the EDIT link on each ViewVC directory page to add/modify the documentation. If an existing documentation has already been provided, then it will be displayed at the bottom of each page. CVS write permission and valid CVS access account is required for editing documentation.
BEE3 Design

- 4 Xilinx FPGA: (FF1136)
  - Virtex-5 LX110T or SX95T
- 16 DIMMs
  - 2 DDR2-400/533/667 channels per FPGA
  - Up to two 4GB DIMMs per channel
- 8 10GBase-CX4 interfaces
- 4 PCI-E x8 slots (endpoints)
- 4 QSH-DP (40 LVDS pairs) daughter card & cable connectors
- 4 GE RJ45 interfaces
- 2U chassis
- ATX12V/EPS2U 500W power supply

Third party to provide manufacturing, distribution, and support.

8/28/2007

Wawrzynek, FPL 2007
For more information …

http://bee2.eecs.berkeley.edu, http://ramp.eecs.berkeley.edu


Thank you!