Designing Network Devices

for Research and Education

Jeffrey Shafer

March 2008
http://www.jeffshafer.com/

Network Systems Architecture

- Networking is essential to modern computer systems
- Networks are continually getting faster
  - Ethernet: 1Gbps, 10Gbps, 40/100Gbps, … (great work by Electrical Engineers!)
- Outstanding challenge today
  - Wire speed is outpacing processing speed

Network Systems Architecture

- Field of **Network Systems Architecture** is actively working to satisfy raw network potential

- Encompasses wide range of network components
  - **Hardware**
    - Network Interface Cards
    - Routers / switches
  - **Software**
    - User Applications (servers applications)
    - Operating Systems (network stack and device drivers)
    - Virtual Machines (multiple operating systems sharing a network)

Network Systems Architecture

- Sending data across a network from New York to LA can interact with all of these components (and more!)
- How to support high data rates?
  - Increase system-level efficiency
  - Increase device functionality
- Many active research areas, including…

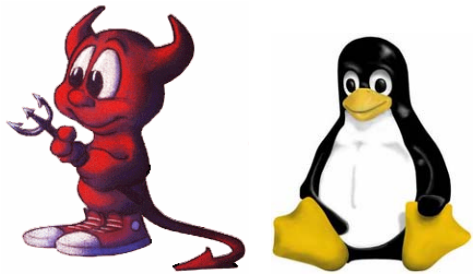
Research Challenges

- **Network Interface Card** design
  - Should we move TCP stack from Operating System to NIC?
  - Should we cache data on NIC?
  - How does the NIC efficiently communicate with software?

- **Router / Switch** design
  - How can we switch packets efficiently at high data rates?

Research Challenges

- **Operating System** design
  - How can we parallelize the network stack and device driver to run on multi-core CPUs?
  - How can we efficiently / flexibly delegate work to the NIC?



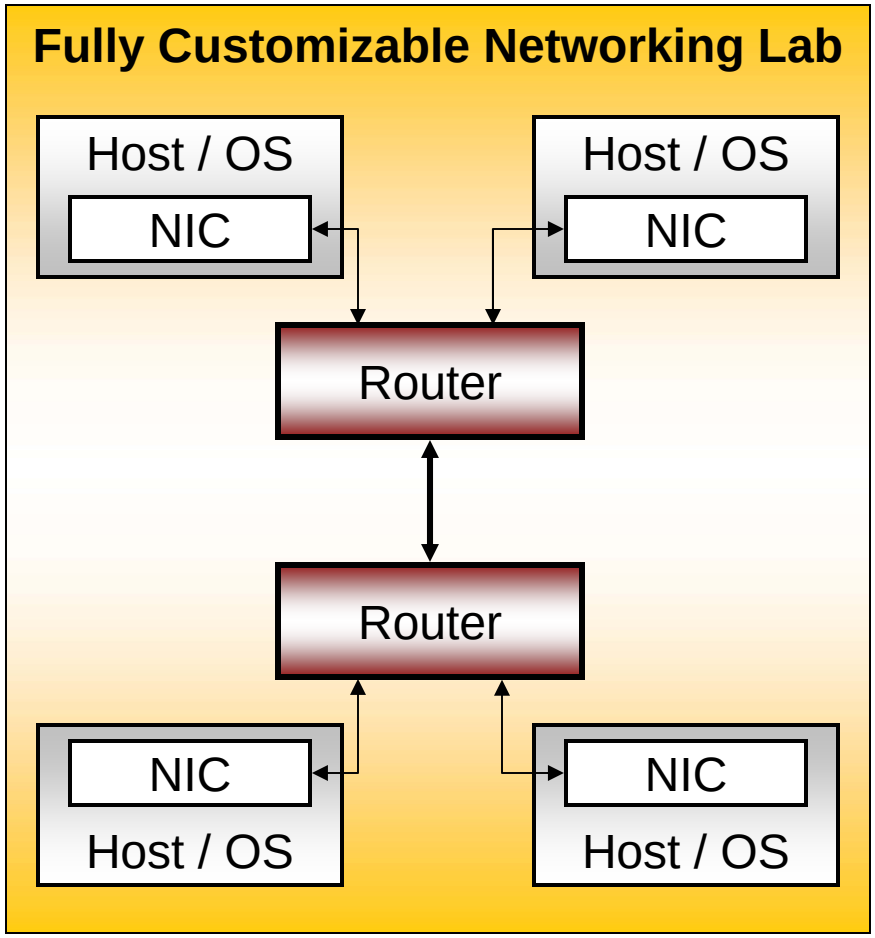
- **Virtual Machine** design
  - How can multiple operating systems on a single computer efficiently share a common network card?

Integrated Research

- Our research group is actively involved in many of these areas
  - Many topics require both hardware and software development in close cooperation
- Lessons learned
  - All elements must cooperate in order to form an efficient network system
  - Future architectural innovations are going to be integrated across components and not isolated
    - e.g. We can't improve the NIC without also improving how the Operating System communicates with the NIC

Integrated Research

- How can we explore an entire network system?
- Need a research environment that allows all major components to be modified experimentally
  - Low Power Networking where router and NIC negotiate for maximum efficiency



Research Strategies

- How can we realize this vision to investigate new network systems architectures?
- Existing tools
  - Simulation with whole-system simulators
    - Maximum flexibility to model any device
    - Challenges when applied to network systems
  - Prototyping with software programmable NICs/routers
    - Better approach, but limited by existing tools
- We have used both approaches in our research projects

Outline

- Existing tools for network systems architecture research
  - Whole-system simulators
  - Prototyping with software programmable devices
- New reconfigurable prototyping platforms
  - RiceNIC
  - NetFPGA
- Research applications
- Education applications

Simulation Challenges

- Network systems require high fidelity modeling
  - Asynchronous interactions between Hardware, Software, and Multiple computing systems
  - Adaptive algorithms (e.g. TCP) sensitive to underlying performance
- Performance
  - 5 orders of magnitude slower (or more!) for cycle-accuracy
  - 120 real-time seconds → 138 days in simulator
    - Long tests needed to reach TCP steady-state performance
  - How do I run hundreds of experiments?
  - How many compute machines do I need for simulation?
- Validation - How do I know my simulator is accurate?

Prototyping Advantages

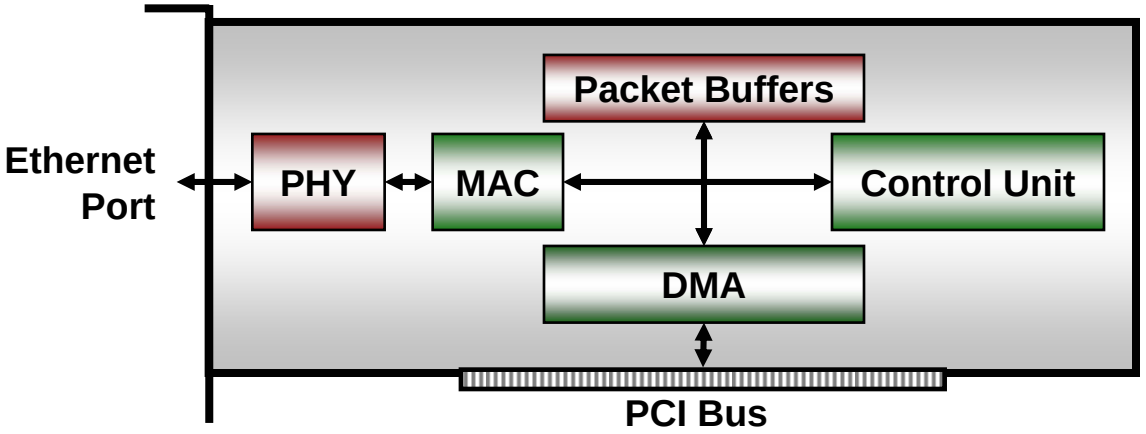
- Performance
  - Prototype can run in real-time, not simulator-time
- Expense
  - Cheaper to build a few prototypes than buy a compute cluster for simulation
- Accuracy
  - A simulator models the entire computer (and has inherent assumptions and approximations throughout the system)
  - A prototype models the specific device being investigated (e.g. NIC) but uses a real system otherwise
    - Experimental error is constrained to the device being studied

Existing Prototyping Options

- Operating System
  - No problem – Use a real one! (Linux)
- Router
  - Hard to modify a Cisco router
  - Build my own?
    - Standard PC with multiple NICs and routing software
- Network Interface
  - Hard to modify your desktop NIC
  - Commercial software-programmable NICs
    - Cluster-computer interconnects (Myrinet, Infiniband)
    - Commodity networks (Ethernet)
- Are these options sufficient to prototype a router or NIC?

NIC Operation

- Central control unit - ASIC
  - Exchanges control descriptors and interrupts with host computer to manage data flow
- Hardware assist units
  - MAC – Transfer data to/from network
  - DMA – Transfer data to/from host CPU
- Packet Buffers – Memory or on-chip FIFOs

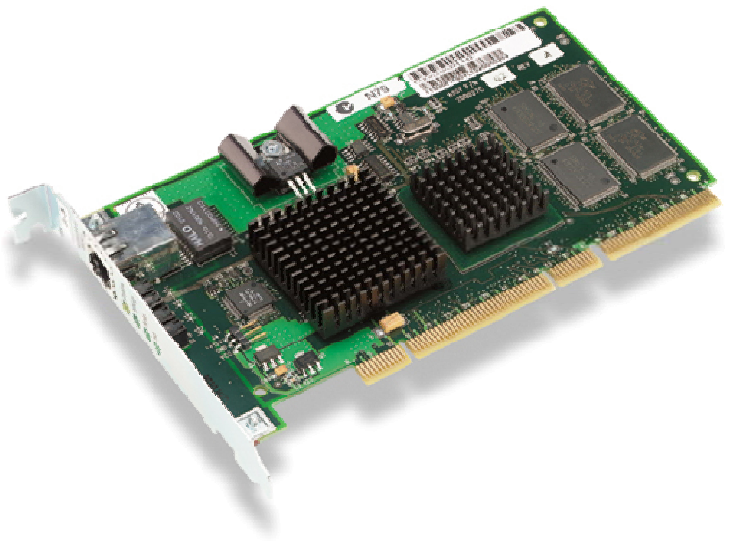


Implemented all in hardware or under flexible software control

Tigon2 Overview

Software-Programmable Gigabit Ethernet NIC

- Embedded processors
  - Two 88 MHz MIPS R4000
- Memory (Control & Packet Buffers)
  - On-chip scratchpad
  - Off-chip SRAM (up to 2MB)
- Hardware assist units
  - MAC – Move data to/from network
  - PCI DMA – Move data to/from host
- All hardware is fixed



Tigon2 as a Prototype

- Tigon2 achieves full gigabit throughput as a NIC
- What happens when we try to add additional functionality for research projects?

- **TCP connection handoff**
  - Move TCP stack to NIC for key connections
  - Throughput limited to 100Mbps by processor and available memory

- **NIC data caching**
  - Save frequent packet data on NIC
  - Cache size severely limited by Tigon2 memory capacity

Existing Platforms Insufficient

- Performance Limitations
- Intellectual Property limitations
  - Unable to obtain existing firmware or technical specifications
  - Restrict range of customizations
- Hardware architecture limitations
  - Hardware and control systems is fixed
  - May constrain new software design
  - Software emulation of a new hardware architecture might be very compute-intensive

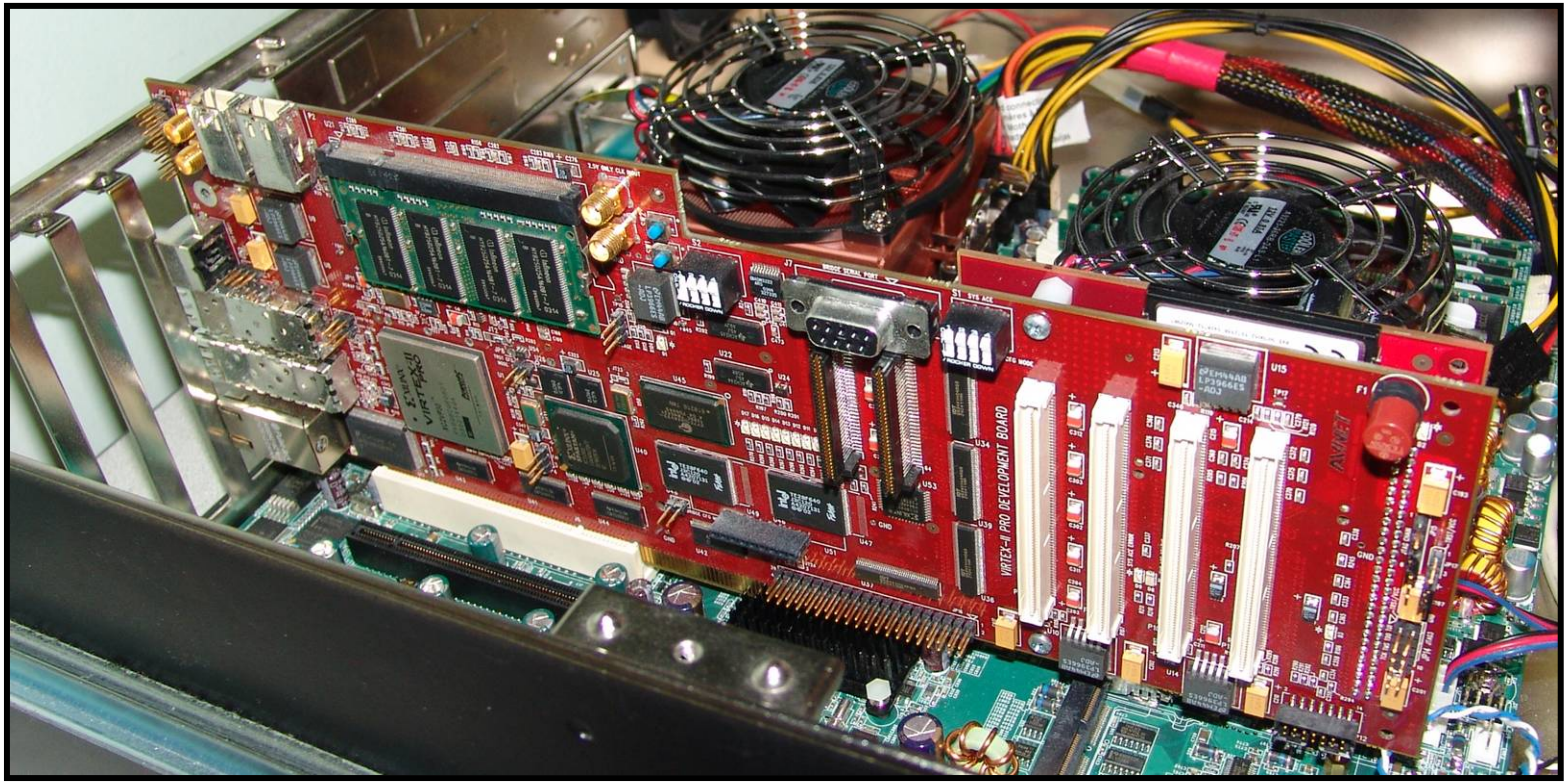
Our prior experiences with experimental prototyping motivated the creation of RiceNIC

Outline

- Existing tools for network architecture research
- Reconfigurable Network Devices
  - RiceNIC – Developed by Rice
  - NetFPGA – Developed by Stanford
  - Development occurred in parallel
  - Both are in active use at Rice for networking research and education
- Research applications
- Education applications

RiceNIC Overview

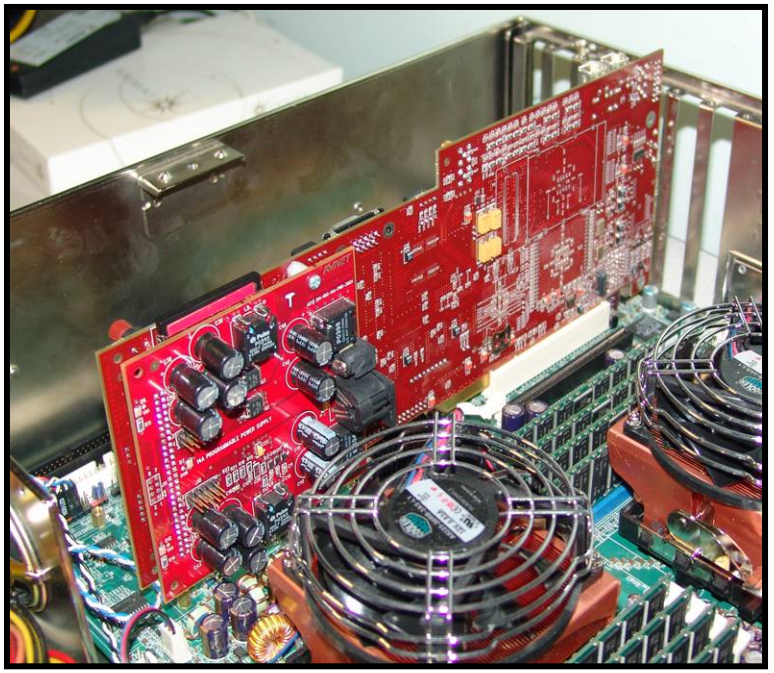
Gigabit Ethernet Network Interface Card



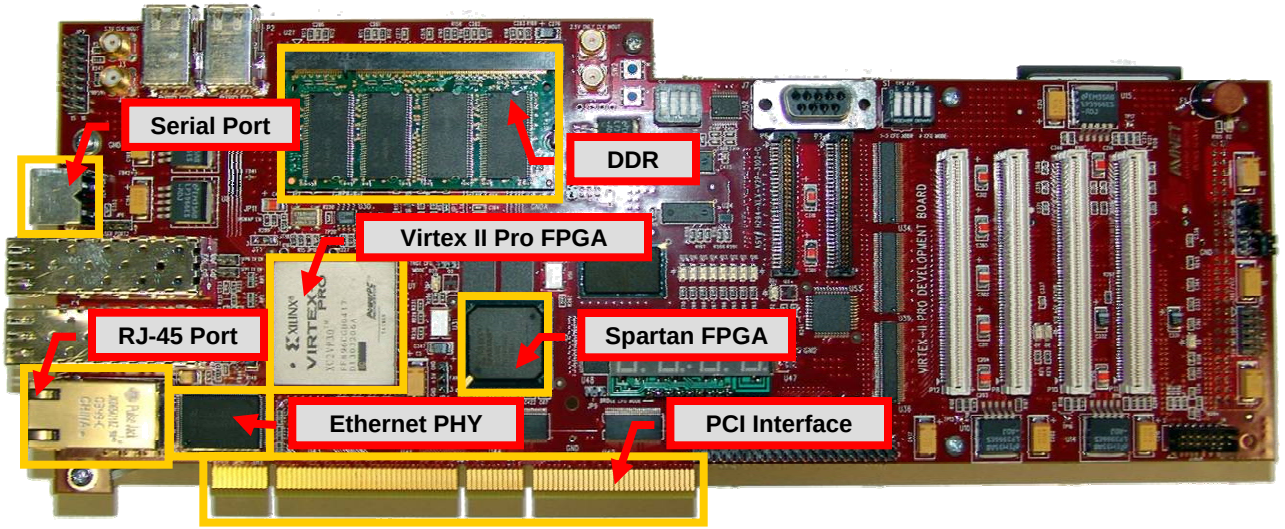
http://www.cs.rice.edu/CS/Architecture/ricenic/

RiceNIC Overview

- Reconfigurable
  - FPGAs implement hardware architecture
- Programmable
  - Embedded processors provide high-level NIC control
- Performs at Ethernet line rate
- Reference design is freely available



RiceNIC Board

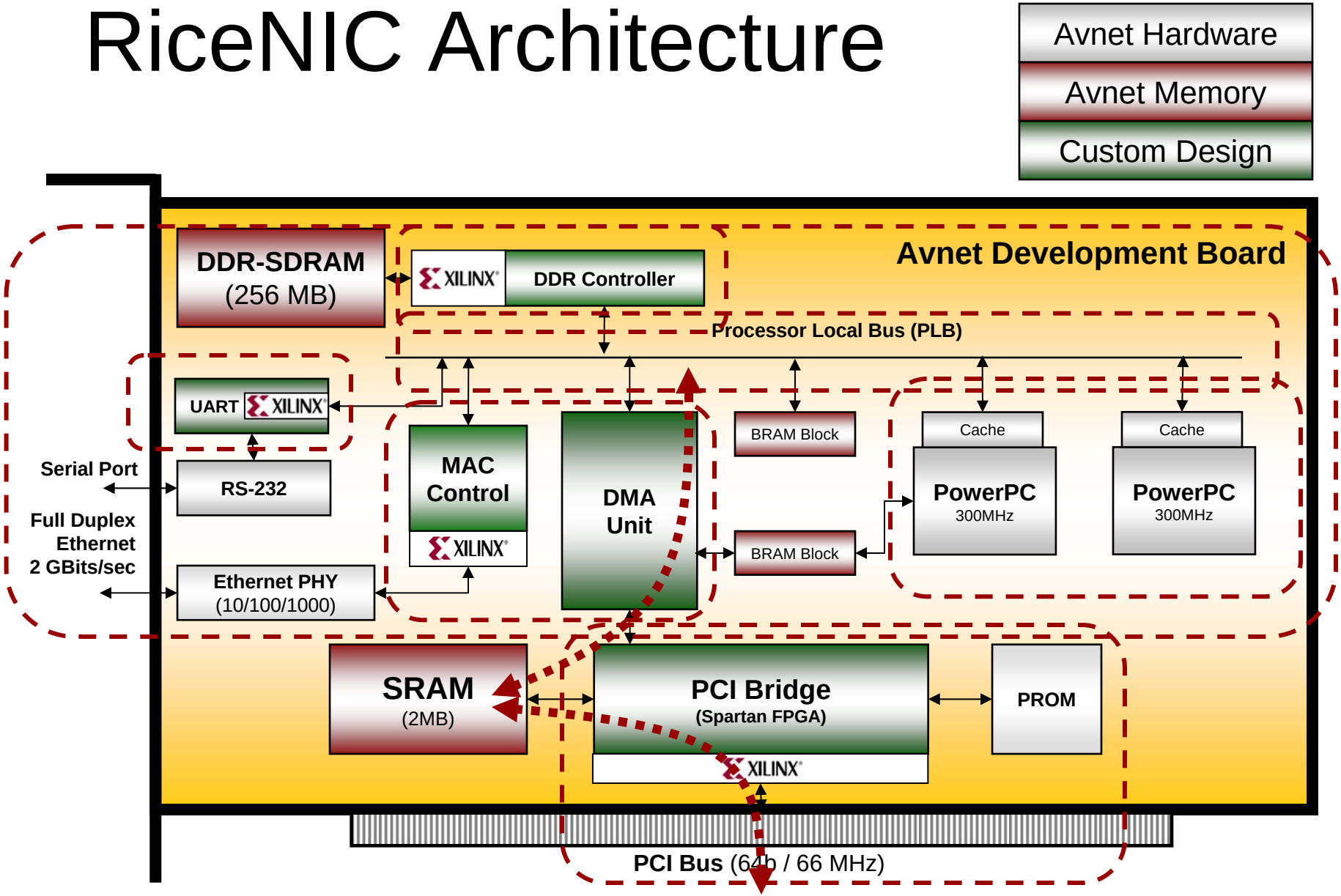


- Avnet Prototyping Board
- 2 FPGAs
  - 2 PowerPC processors (300 MHz)
- Serial Port
- On-NIC Memory (256MB DDR)
- Gigabit Ethernet PHY
- 64-bit / 66 MHz PCI bus

RiceNIC Operation

- Generic NIC components
  - Processing core: ASIC or custom processor
  - MAC and PCI transfers: Dedicated logic
  - Packet Buffering: Small SRAM
- What parts of RiceNIC correspond to these functions?
  - Processing: Custom logic on FPGAs and firmware on general-purpose PowerPC processor
  - MAC and PCI transfers: FPGA logic
  - Packet Buffering: Large DDR SODIMM

RiceNIC Architecture

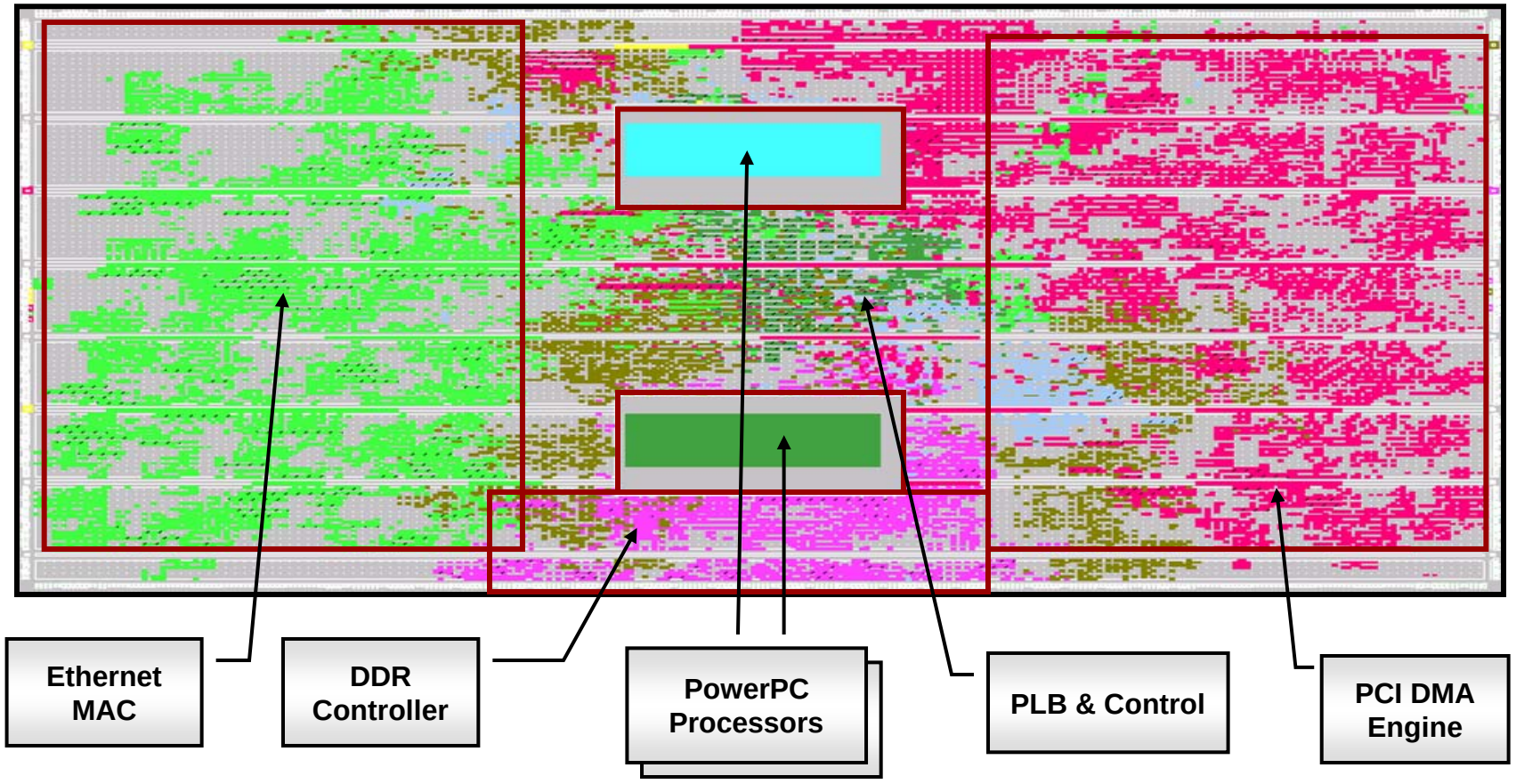


FPGA Utilization

- Implemented with Xilinx tools (ISE and EDK)
- Used ChipScope on-chip logic analyzer
- Spare capacity for researchers to prototype new systems

Component	Virtex FPGA		Spartan FPGA	
Slice Flip Flops	9,089 / 27,392	33%	2,361 / 6,144	38%
4 input LUTs (Logic)	11,811 / 27,392	43%	2,504 / 6,144	40%
BRAMs	51 / 136	37%	6 / 16	37%
Occupied Slices	9,164 / 13,696	66%	3,070 / 3,072	99%
Global Clocks	10 / 16	62%	2 / 4	81%
Digital Clock Managers	5 / 8	62%	N/A	N/A

Virtex FPGA Placement



RiceNIC Debugging Tools

- Software Debugging
  - Serial Console (RS-232 port)
    - Command line interface to PowerPC processors
    - Runtime debugging / configuration changes / bootstrapping
  - Firmware Profiler
    - Timer based statistical profiler (similar to Oprofile)
    - Exports results via serial console
- Hardware Debugging
  - Logic analyzer (Xilinx Chipscope) on FPGAs
  - Pin headers, 7-segment display, JTAG

Debugging Tools are Essential for Experimentation

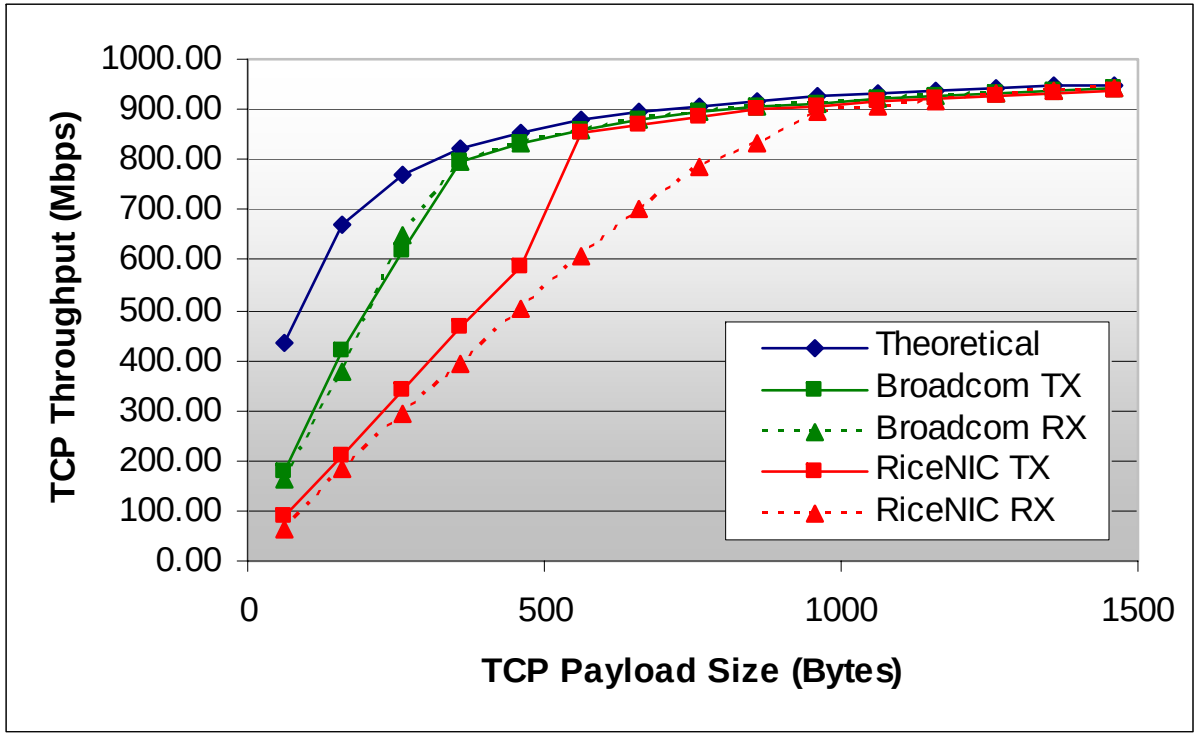
RiceNIC Design Timeline

- Development time
  - Hardware (FPGA) design - 1 year
  - Software (firmware / driver) design – 1 month (with significant past experience)
- Development stages
  - Learning Xilinx FPGA tools – 1 month
  - Design / Implementation – 9 months
  - Testing – 3 months

Constructing an experimental prototype is practical!

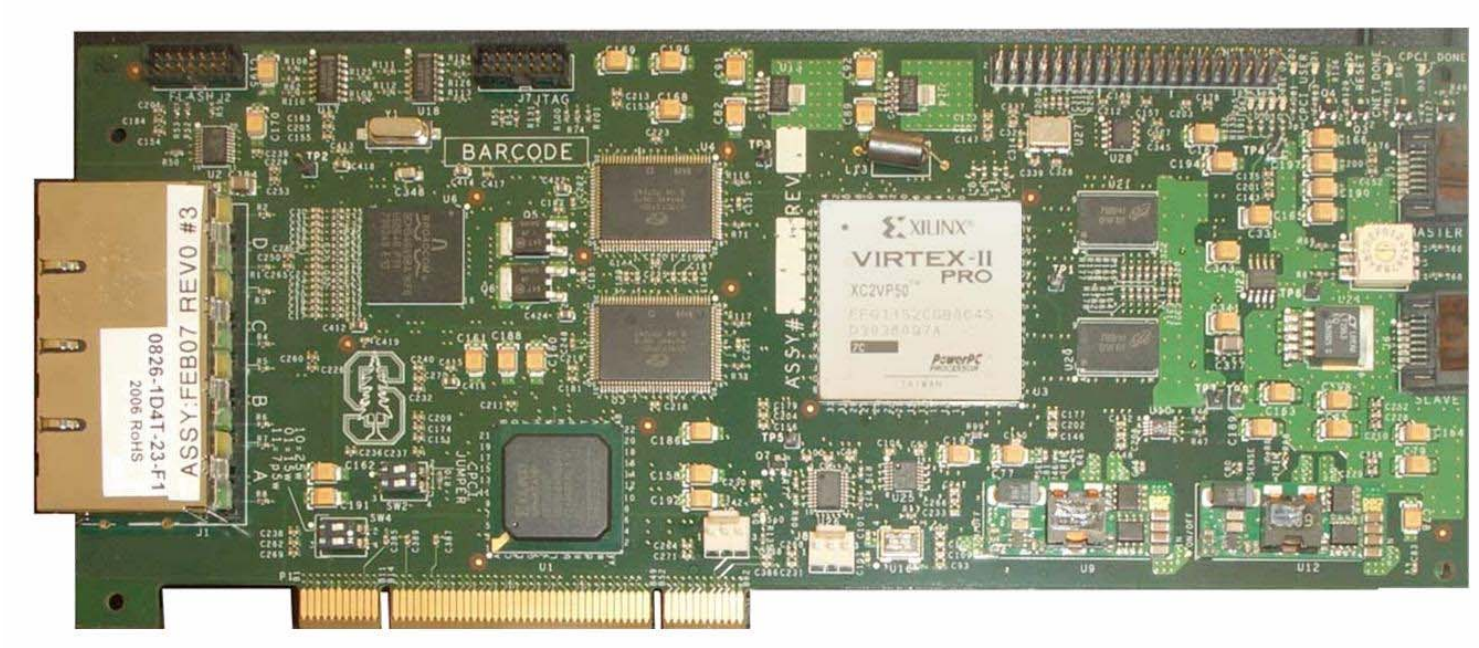
RiceNIC Performance

- TCP streaming test (netperf)
- High throughput for packets >1000 bytes



NetFPGA Overview

4-Port Gigabit Ethernet Router

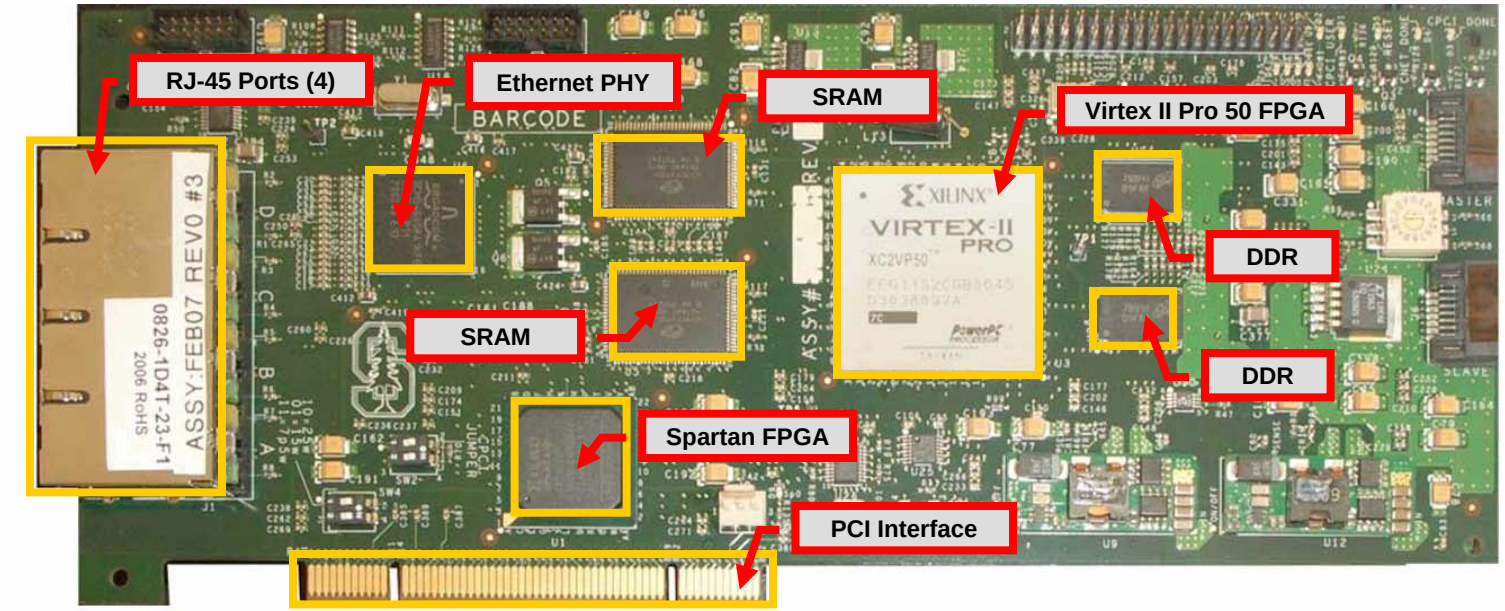


http://netfpga.org

NetFPGA Overview

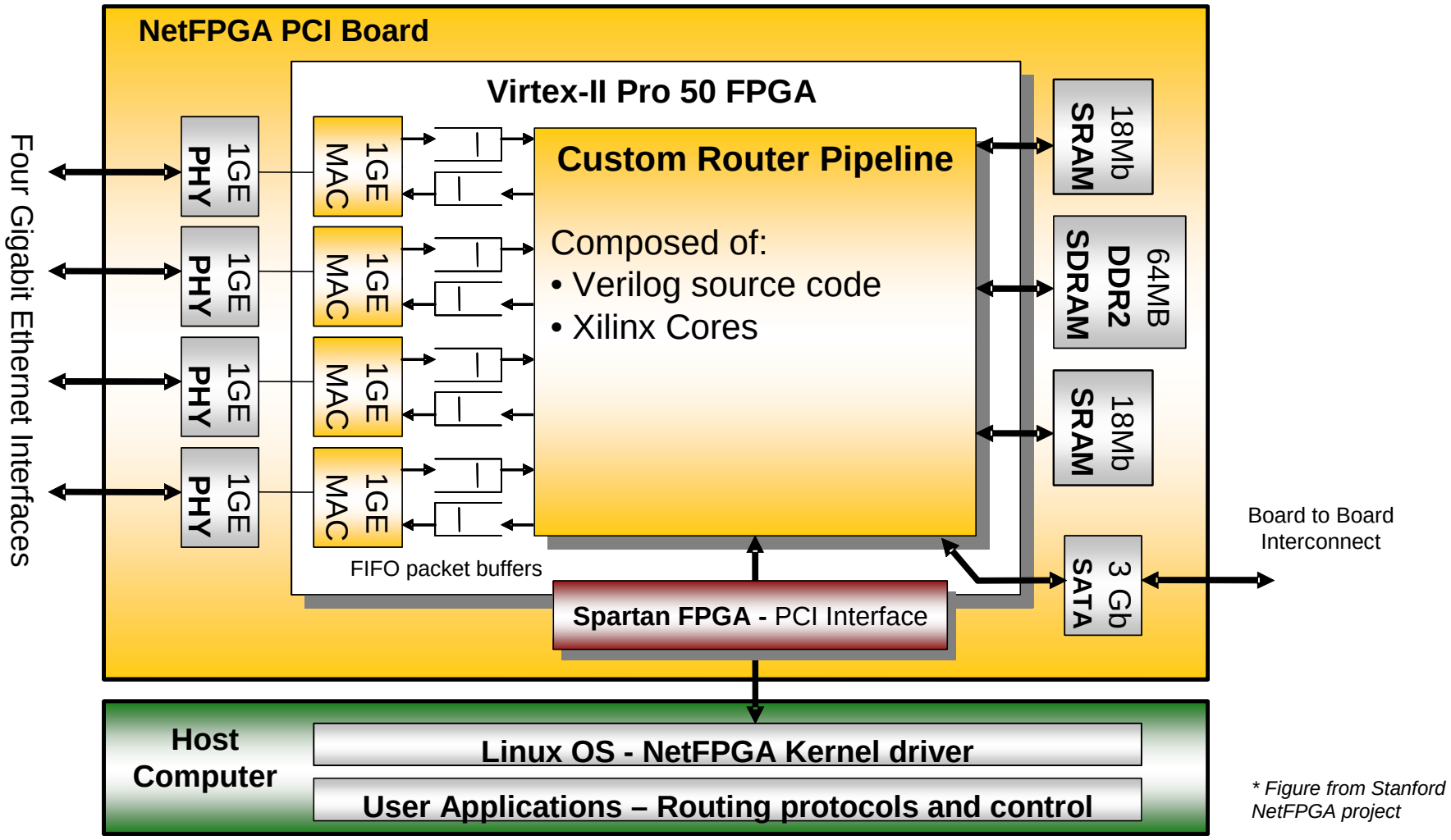
- **Router Board**
  - Ethernet Ports
  - Hardware fast-path for packet forwarding
- **Host PC**
  - Linux OS
  - Runs high-level software (e.g. router-to-router control protocols)

NetFPGA Board



- Custom Development Board
- 2 FPGAs
  - 2 PowerPC processors (not used)
- 4 Gigabit Ethernet ports
- On-Board Memory
  - 4.5MB SRAM / 64 MB DDR2
- Custom Verilog router design
  - Fast-path for packet forwarding
- 32-bit PCI Interface

NetFPGA Architecture



** Figure from Stanford NetFPGA project*

Similarities – RiceNIC and NetFPGA

- Project development occurred in parallel
- Common objectives
  - Design flexibility for wide variety of research and education applications
- Significant similarities
  - Use of FPGAs as prototyping platform
  - Hardware resources: DDR, SRAM
  - Open source code to enable design modifications
  - Design partitioned between software and hardware components

Different Project Motivations

- NetFPGA
  - 4-port router
  - Primary goal: **Education**
  - Current design is 3rd revision of their platform, and is flexible enough for some research applications
- RiceNIC
  - Ethernet NIC
  - Primary goal: **Research**

Different Design Choices

NetFPGA

- 32-bit PCI bus
- Host CPU (general purpose)

RiceNIC

- 64-bit PCI bus
- Two embedded PowerPC processors

- Implications of processor choice for complex software?
- Host CPU
  - Loose coupling with hardware board – Slow (high latency)
  - Plentiful resources and full-featured development tools
- Embedded processors
  - Tight coupling with hardware – Fast!
  - Resource (memory) constrained and challenging to debug

Different Development Strategy

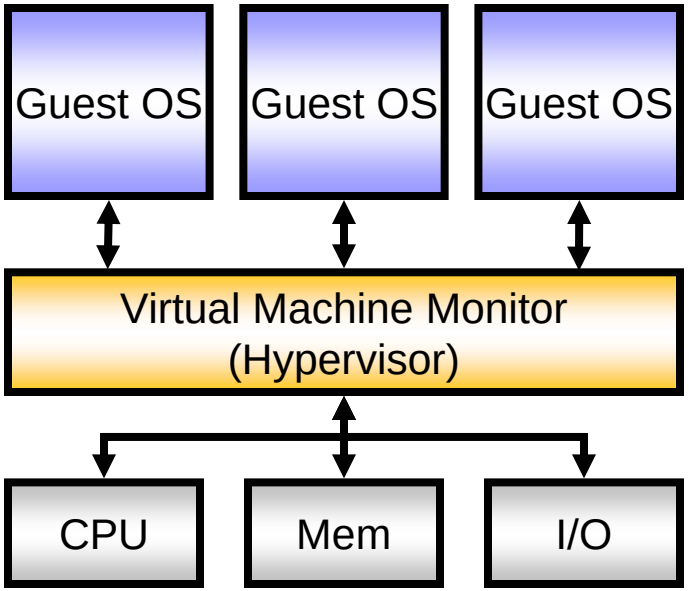
- NetFPGA – Custom board
  - Highly customized to specific needs
  - Expensive / time consuming (did receive industry donations and technical assistance)
  - Still have to do FPGA design after producing the board!
- RiceNIC – Commercial FPGA prototyping board
  - Inexpensive to obtain
  - Board available immediately for development
  - At mercy of commercial vendors to keep manufacturing board!

Outline

- Existing tools for network architecture research
- Reconfigurable Network Devices
  - RiceNIC and NetFPGA
- Research applications
  - Virtual Machines
  - Low-Power Networking
- Education applications

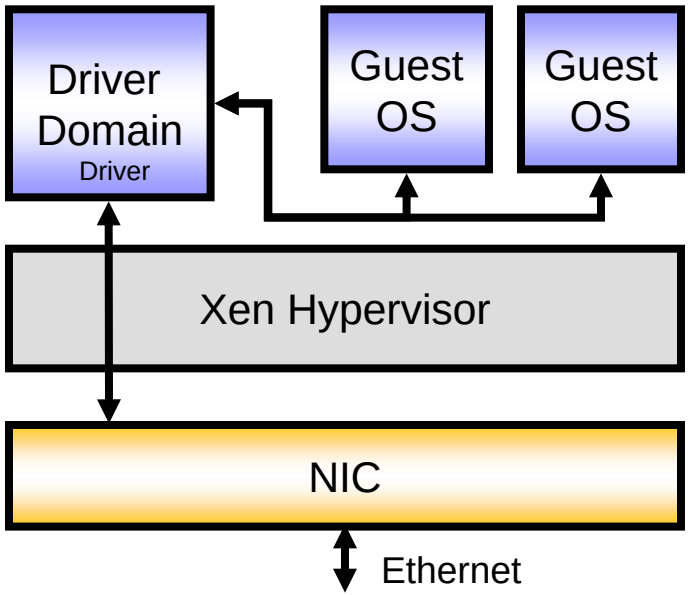
Virtualization

- Run multiple operating systems on a single machine
  - Application: Server consolidation
- Virtual machine monitor (VMM)
  - Provides abstract hardware interface
  - Shares hardware resources (CPU, memory, I/O)
  - Switches between active guest domains
- Popular VMMs
  - VMware
  - Xen (open source)



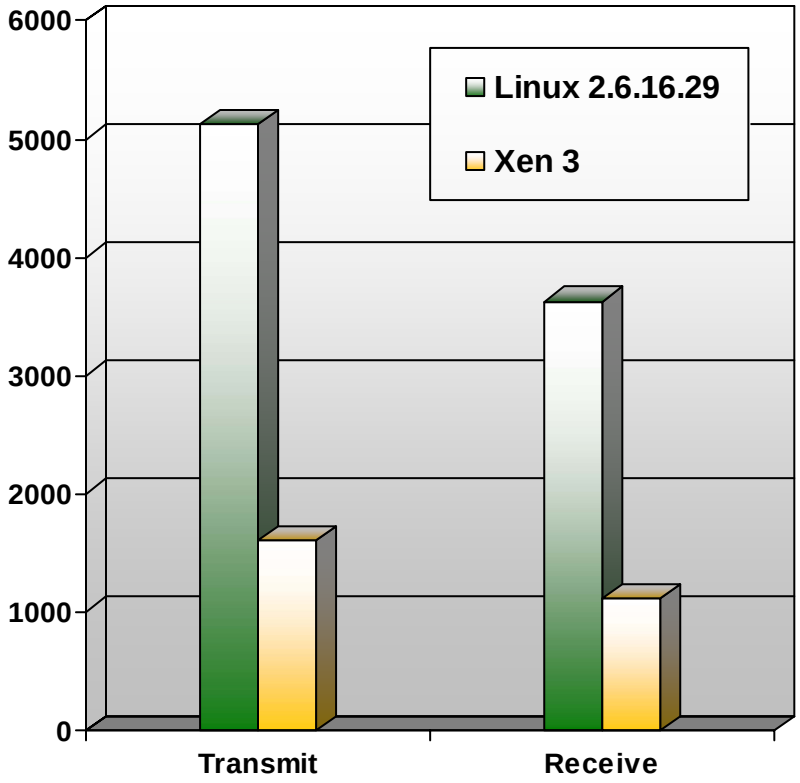
Networking in Virtual Machines

- Conventional NIC must be shared in software between multiple guests
  - Multiplexing / demultiplexing
- Xen approach
  - Driver domain runs device driver and communicates with NIC
  - Guest OS's communicate through driver domain



Xen 3 Network Performance

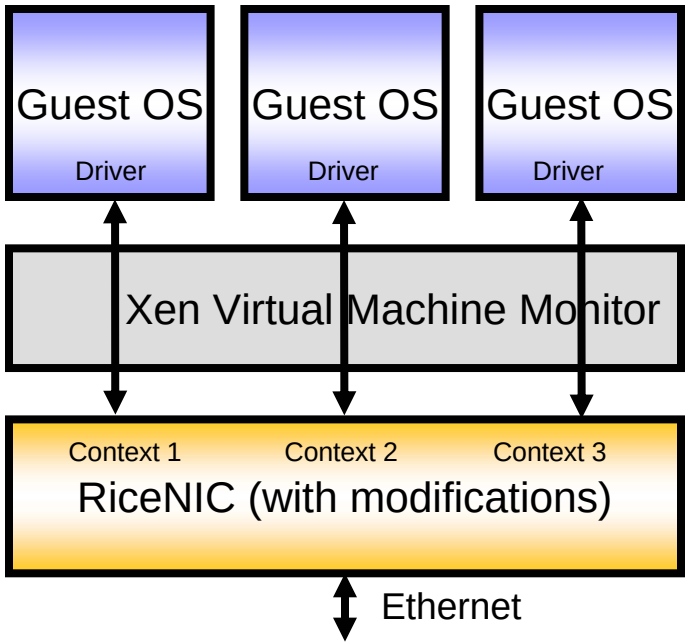
2.4GHz Opteron



- Single guest performance shown
  (Multiple guests even worse!)
- Virtualization incurs significant overheads
  - Software bridge multiplexes data to share NIC
  - Context switching between guests and driver domains
  - Copying packets in the hypervisor to move between domains

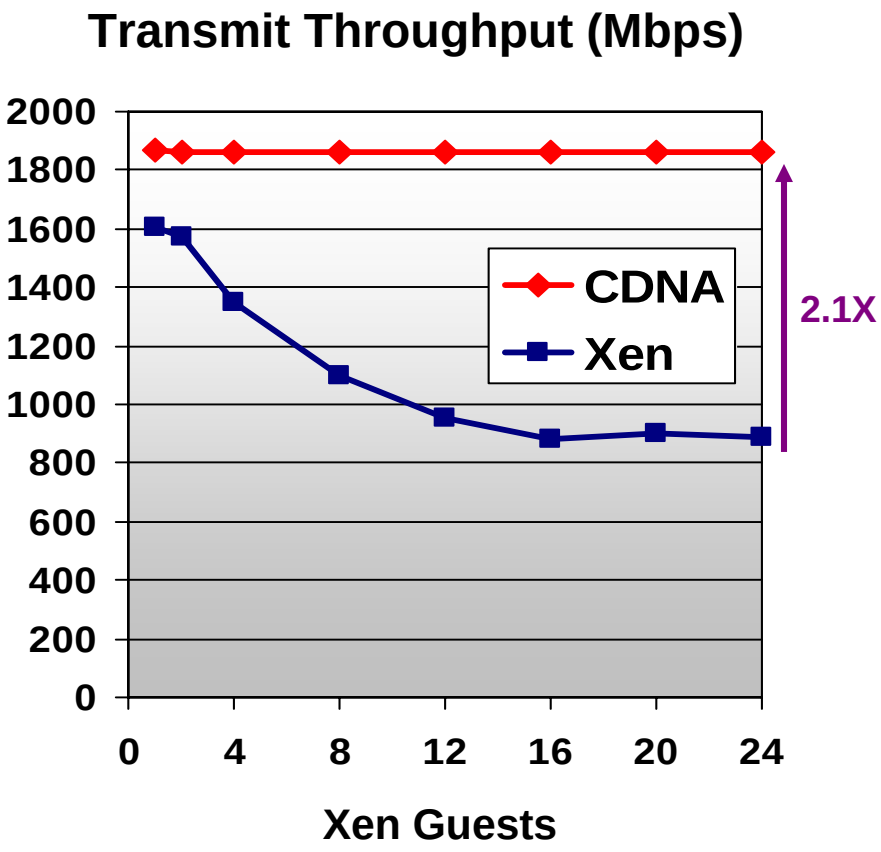
Concurrent Direct Network Access †

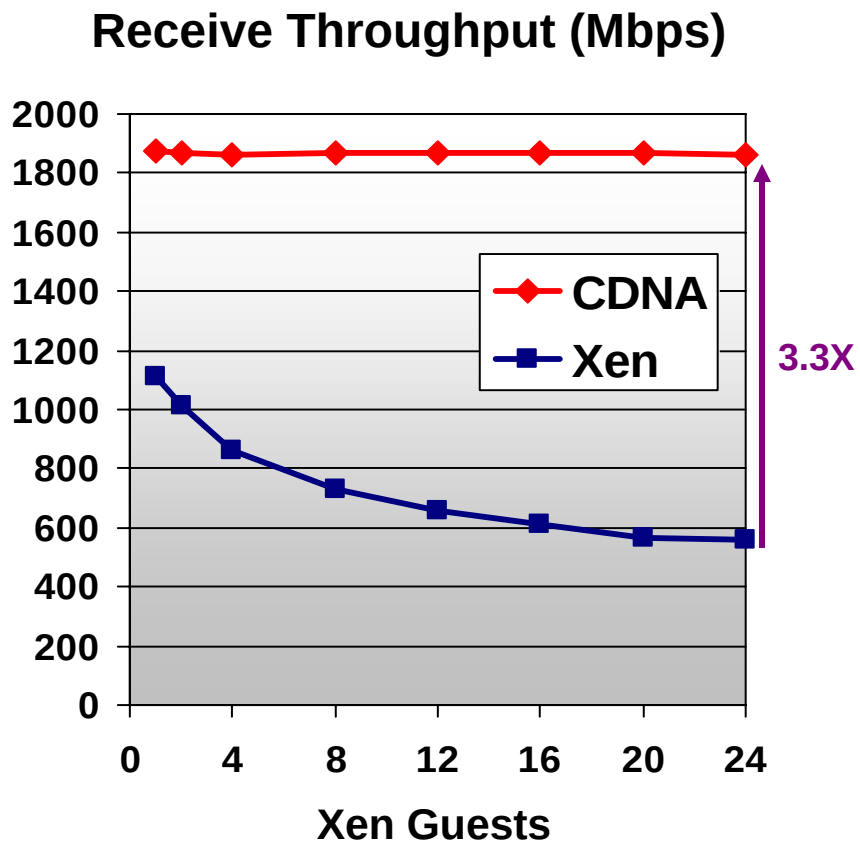
- Goal: Improve virtualization performance by making NIC more intelligent
- Each guest OS can talk directly to single shared NIC through separate interfaces
- Performance improved due to reduced CPU load on host system
  - No need for Virtual Machine to multiplex / demultiplex packets
  - No need to context switch between guest OS and driver domain



† P. Willmann, J. Shafer, et.al, Concurrent Direct Network Access for Virtual Machine Monitors, *The International Symposium on High Performance Computer Architecture (HPCA'07),* Phoenix, AZ, (Feb 2007)

CDNA Performance Scaling





Architecture Modifications

- Optimized hardware and software together
- RiceNIC FPGA
  - Isolated contexts in memory
  - Hardware event notification
- RiceNIC software
  - Firmware provides packet multiplexing / demultiplexing
    - Data locality – 300MHz PowerPC outperforms 2.4GHz Opteron!
  - Used 1 PowerPC processor and 12MB of RAM
- Xen / OS modifications

Design highlights the improvement possible by treating the network as a system

Benefits of RiceNIC

- Prototyping (with RiceNIC) critical to success of virtualization research
- Could not use software programmable NIC
  - Needed to change hardware architecture
  - Software emulation too slow
- RiceNIC prototype ran at full speed
- Used RiceNIC to experimentally determine key architectural features
  - Minimum on-NIC packet buffer size per virtual machine
- Could not obtain equivalent results via simulation in a timely fashion

Low Power Networking

- Energy Efficient Internet Project † *(at USF and UF)*
  - Can a new power-aware MAC controller reduce the power consumption of NICs?
- Modified RiceNIC and NetFPGA hardware
  - Replaced MAC core on FPGA with a custom low-power variant that supports adaptive link rates

† The Energy Efficient Internet Project: http://www.csee.usf.edu/~christen/energy/main.html

Low Power Networking

- This research could not be done on any software-programmable NIC!
  - MAC core is a fixed chip on traditional NICs
  - FPGA allows multiple MAC designs to be evaluated experimentally
- Future possibilities
  - NIC *(RiceNIC)* and router *(NetFPGA)* communicate to determine optimal power level
  - NIC, OS, and applications communicate to determine when to transition from low-power to high-power mode based on future data transfer needs

Outline

- Existing tools for network architecture research
- Reconfigurable Network Devices
  - RiceNIC and NetFPGA
- Research applications
- Education applications

Education

- Reconfigurable Network Devices are essential for future innovation
- What are the implications of this?
  - These devices are complicated systems with hardware and software components
  - Need to teach Computer Engineering students how to construct them!
  - Best approach: Project-based classes
  - Exciting new opportunities for networking courses with integrated hardware / software components

Education with NetFPGA

- New course at Rice – COMP/ELEC 519
  - Network Systems Architecture
- Use NetFPGA system in group projects to develop
  - Software IP Router *(C programming)*
  - Hardware Ethernet Switch *(Verilog programming)*
  - Hardware IP Router + Software Control
- Good capstone Computer Engineering course
  - Groups must become familiar with both hardware and software elements of their system to succeed

Education with RiceNIC

- Similar potential for FPGA hardware-oriented projects
- Additional capabilities due to embedded processors
- Example class project
  - Added Network Address Translation module to RiceNIC firmware
  - Project time – 1 day (suitable for a lower-level course)
  - NIC still runs at full Ethernet speeds
- Could have used a commercial software-programmable NIC for this project, but…
  - Debugging of NAT module greatly assisted by RiceNIC serial console which printed packet traces

Education

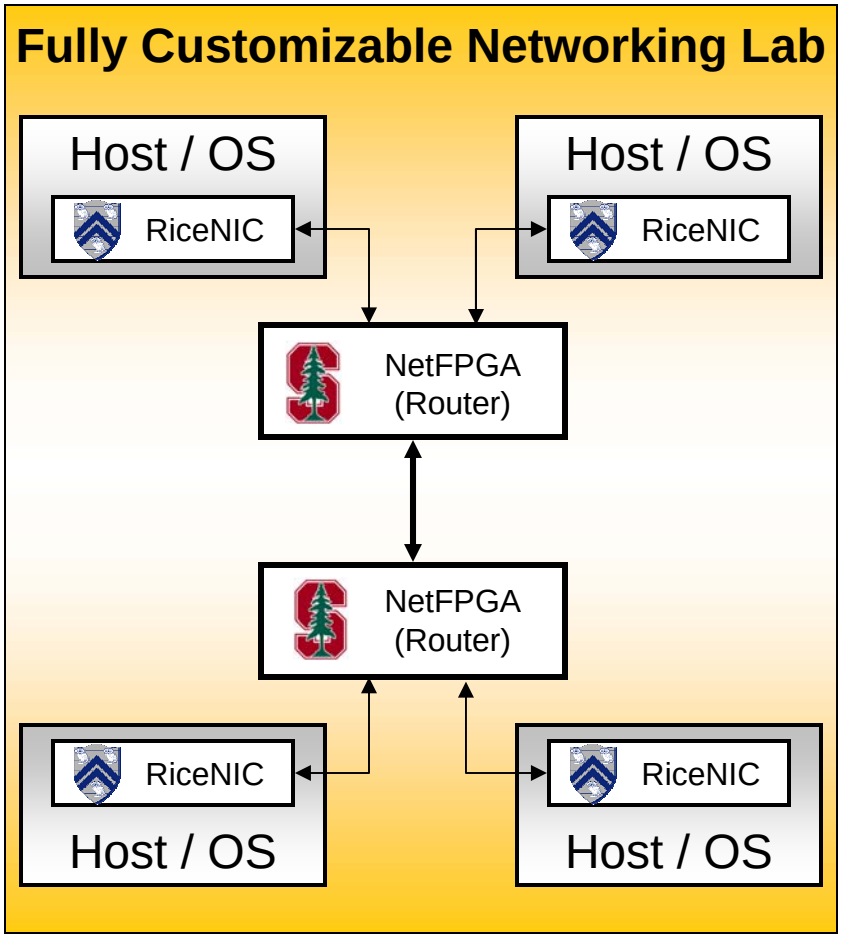
- Computer Engineering skills provided by all these projects transcend networking
  - Why networking?
    - Important and exciting topic
    - Useful vehicle for teaching these larger skills
- Prototyping valuable to education, not just research
  - Provides hands-on experience
  - Simulation too slow to evaluate systems
    - Valuable to have the students run 50 shorts tests on a prototype, pick the best solution, and defend it

Conclusions

- Research into network systems demands experimental prototypes
  - Complex, asynchronous interactions among system components
  - Simulation insufficient for new architectures
- **Reconfigurable Network Devices** enable research and education that could not otherwise be performed

Conclusions

- RiceNIC and NetFPGA are two examples of reconfigurable network devices
- Can be used in fully customizable networking lab that allows experimental exploration of new networking architectures



Acknowledgements

- **RiceNIC Development**
  - Faculty: Scott Rixner
  - Students: Hyong-Youb Kim, Tinoosh Mohsenin, Jeff Shafer
- **RiceNIC Applications**
  - Faculty: Alan Cox, Scott Rixner, Willy Zwaenepoel
  - Students: David Carr, Michael Foss, Hyong-Youb Kim, Kaushik Kumar, Aravind Menon, Jeff Shafer, Paul Willmann
- **NetFPGA Development**
  - Faculty: Nick McKeown, John Lockwood
  - Students: Adam Covington, David Erickson, Glen Gibb, Paul Hartke, Brandon Heller, Jad Nauos, Greg Watson, Jim Weaver
- **NetFPGA Applications**
  - Currently in Beta Release

Questions?

Designing Network Devices
for Research and Education

March 2008

RICE