Teaching Networking Hardware

Stanford High Performance Network Group

Presented By: Martin Casado
Gregory Watson
Martin Casado
Nick McKeown

June, 2005
Motivation
(networking hardware experience needed in classroom)

- Many students go to work in the networking industry
  - complex hardware-software systems
  - high speed, asynchronous network environments

- Undergraduate (and graduate) hardware classes typically focus on processor design
  (... how many graduating students design processors?)

- Want to create practical, useful environment for developing working network systems
Motivation
(cool stuff in networks is hard to get to)

\[ \{ \text{App, TCP/IP, Phy} \} \quad \text{OS + hardware = hard to teach} \]
(The Punchline)
CS344: Advanced Projects in Networks

- Students design routers in hardware and software
- Teams of 1 hardware student, 1 software student
- Routers must route live Internet traffic
- All routers must interoperate on a complex topology
- Students then get to “show off”
Presentation Overview
(a bit late)

- Technologies developed to support class
  - NetFPGA
  - VNS

- Class Overview

- Experiences in Classroom (2004, 2005)

- Looking Ahead
Course Tools

- **NetFPGA**: a programmable 8-port Ethernet device that can be programmed and tested remotely.

- **VNS**: Allows user space processes to participate as routers on the Internet.

- **VNS + NetFPGA**: Used to develop, test and deploy hardware+software routers on arbitrary topologies connected to the Internet.
NetFPGA

• Programmable Network device

• Each board contains three FPGAs and an 8-port Ethernet controller.

• A suite of scripts that invoke the various design tools (industry standard)

• Libraries to make it easy to create verification scripts that run in simulation environment as well as on the actual hardware.

• A web interface for remotely downloading and running the designs.
NetFPGA Board
Virtual Network System
(emulate network topologies)
Virtual Network System

- Maps NetFPGA boards into complex networks connected to the Internet

- Allows students to write software “cpu” for boards as user space programs (in C)

- Connect standard servers (e.g. Linux + apache)

- Currently used in undergraduate networking courses to build software only routers (stanford, WSU, Johns Hopkins)
VNS + NetFPGA =

• Build hardware-software routers on the Internet

• Each team can have an isolated topology or multiple teams can connect to the same topology (interoperation)

• Entire design and development process can be done remotely (e.g. Remote University) (students never have to see an actual NetFPGA board!)
CS344: Building an Internet Router

Software

- CLI
- “User Apps”
- TCP
- IP/ICMP/OSPF

Hardware

- ARP
- IP Forwarding
- Layer 2 forwarding
CS344: Building an Internet Router

- Build full router in hardware and software (6 weeks)
  - Route live Internet traffic
  - Managed via CLI
  - Generate forwarding tables using OSPF-like protocol
  - Converge in complex topology after link failure

- Inter-operate with all other routers in class on complex network topology

- Open-ended design component (build something neat) (2 weeks)

- Comprehensive design, implementation and testing documentation
Hardware Requirements

- Layer-2 switching
- Protocol decoding
- ARP cache lookup
- Forwarding
  - Longest prefix match
  - Decrement ttl
  - Calculate IP checksum
- Handle read/write register packets from software
Software Requirements

• ARP  (request/response/cache)
• ICMP  (echo, port unreach, ttl timeout)
• TCP Stack (provided)
• CLI
  • ping/traceroute
  • Similar to IOS
  • Manage interfaces, forwarding table etc.
• OSPF-like protocol
• Control hardware by reading writing registers using special packets
Topologies for Testing
(per team)

Plus any other requested topology
(popular during advanced functionality)
CS344

Interoperability

- Organized by the students (entirely)
- Must be able to converge before tcp timeout during link failure
- For final test, we randomly sprinkle topology with students routers and see what happens!
Advanced Functionality

- Open-ended (almost anything goes)
- 2 requirements
  - Something neat
  - Hardware + software component
- Must consider hardware/software tradeoffs during design
- Present designs in front of panel of industry experts
Classroom Experience

- 2004: 7 students  (3 teams)
- 2005: 18 students  (6 teams)
- Only 1 team has failed to produce working router
- Software typically 12,000 – 20,000 lines of C
- Hardware 7,000 – 11,000 lines of verilog
- Easy week = 10 – 15 hours of work
- Hard week = 40 – 60 hours of work
- Student evaluations very positive
- Industry feedback very positive
Classroom Experience

- Hardware/software students must learn how to communicate
- Interoperation is hard for students (many, many edge conditions)
- Design is really important ... bad start = chaos later on
- Software bug fix = a few minutes
- Hardware bug fix = a few hours
Classroom Experience

Advanced Functionality

- MAC level encryption (DES)
- SSH man in the middle attack
- Intrusion Detection System
- Random Early Detection
- Token-based rate limiting
- Firewall
- NAT/VPN
- Distributed DNS cache

- Have really impressed industry panel
And Now ...

- Developing NetFPGA2
  - PCI cards
  - 1 Gb/s
  - On board CPU
- Working on curriculum for introductory architecture course
- Would like to see NetFPGA used outside of Stanford
- If you are interested, please contact us directly!
  - casado@cs.stanford.edu
  - http://yuba.stanford.edu/vns
  - http://yuba.stanford.edu/netfpga