Teaching Network Systems Design With Network Processors: Challenges And Fun With Networking

Douglas Comer

Computer Science Department
Purdue University
250 N. University Street
West Lafayette, IN 47907-2066

http://www.cs.purdue.edu/people/comer

PART I

Introduction

Topic And Scope

Network processors in academia: graduate and undergraduate curricula, lab facilities, and projects

Plan For The Talk

- Introduction and overview
- An example graduate course
- An example undergraduate course
- Example lab facilities
- Discussion

Why Should Your Institution Teach Network Processors?

- Exciting new topic
- Popular among students
- Access to state-of-the-art technologies
- High teaching evaluations
- Just plain fun

Why Should Your Institution Teach Network Processors? (continued)

- Gain familiarity with emerging technology
- Expose students to new hardware and programming paradigms
- Force students to think about design of network systems
- Allow students to experiment with embedded systems
- Prepare students for research

Possible NP Course Emphasis

- Hardware engineering
  - Internal structure of network processor chip(s)
  - Design of external interfaces
  - Engineering tradeoffs
- Software design
  - Programming models and paradigms
  - Special-purpose programming languages
  - Approaches to parallelism

Possible NP Course Emphasis (continued)

- Networking
  - Analysis of protocols
  - Implementation of a stack
  - Monitoring and control of traffic
- Network systems design
  - Overall design of switch, router, firewall, etc.
  - High-speed protocol implementation
  - Integration of hardware and software

Possible NP Course Emphasis
(continued)

- Networking
  - Analysis of protocols
  - Implementation of a stack
  - Monitoring and control of traffic
- Network systems design
  - Overall design of switch, router, firewall, etc.
  - High-speed protocol implementation
  - Integration of hardware and software

What Should You Tell Students?

- There is a huge opportunity for
  - Learning a new technology
  - Research in a new field

The Challenge

Discover ways to improve the design and manufacture of complex networking systems.

The Big Questions

- What systems?
  - Everything we have now plus new
- What physical communication mechanisms?
  - Existing and emerging communication technologies
- What speeds?
  - Two orders of magnitude beyond those in use
- What protocols?
  - Traditional and new

The Big Questions
(continued)

- What applications?
  - New applications not yet designed / standardized

The Challenge (restated)

Discover flexible, general technologies that enable rapid, low-cost design and manufacture of scalable, robust, efficient network systems that run existing and new protocols, perform existing and new functions for higher-speed networks to support a variety of applications.

Special Difficulties

- Ambitious goal
- Vague problem statement
- Problem is evolving with the solution
- Pressure from
  - Changing infrastructure (e.g., wireless)
  - Changing applications (e.g., VoIP)

Statement Of Hope
(1990 version)

If there is hope, it lies in faster CPUs.

Statement Of Hope
(1995 version)

If there is hope, it lies in ASIC designers.

Statement Of Hope
(1999 version)

???

If there is hope, it lies in ~~ASIC designers~~.

Network Processors To The Rescue

- Devise new hardware building blocks
- Make them programmable
- Include support for protocol processing and I/O
  - Embedded processor(s) for control tasks
  - Special-purpose processor(s) for packet processing tasks
- Provide hardware for specialized tasks such as table lookup
- Integrate as much as possible onto one chip
- Call the result a *network processor*

Definition

A network processor *is a special-purpose, programmable hardware device that combines the low cost and flexibility of a RISC processor with the speed and scalability of custom silicon (i.e., ASIC chips). Network processors are building blocks used to construct network systems.*

Statement Of Hope
(2003 version)

programmers!

If there is hope, it lies in ~~ASIC designers~~.

Disclaimer

In the field of network processors, I am a tyro.

Definition

Tyro *Ty'ro*\\, n.; pl. *Tyros*. A beginner in learning; one who is in the rudiments of any branch of study; a person imperfectly acquainted with a subject; a novice.

By Definition

In the field of network processors, you are all tyros.

In Our Defense

When it comes to network processors, everyone is a tyro.

Questions?

PART II

An Example Graduate Course On Network Processors

Goals

- Become familiar with a variety of network processor architectures
- Be able to assess and discuss design tradeoffs and limitations of each approach
- Learn the details of at least one NP
- Gain experience implementing protocol processing functions in software
- Understand the issues of scaling a network system

Organization

- Seminar
- Professor
  - Gives a few introductory lectures
  - Leads discussion
  - Asks questions
- Students
  - Read about network processors
  - Design and implement a project
  - Report on their project

Topics

- Hardware architectures for protocol processing
- Classification
- Switching fabrics
- Traffic management
- Network processors
- Design tradeoffs and consequences
- Details of one example network processor
  - Programming model and program optimization
  - Cross-development environment

Topics

- Hardware architectures for protocol processing
- Classification
- Switching fabrics
- Traffic management
- Design tradeoffs and consequences
- Details of one example network processor
  - Programming model and program optimization
  - Cross-development environment

What Students Do Not Learn

- EE details
  - VLSI technology and design rules
  - Chip interfaces: ICs and pin-outs
  - Waveforms, timing, or voltage
  - How to wire wrap or solder
- Economic details
  - Comprehensive list of vendors and commercial products
  - Price points

Example Grading Scheme

- Preliminary project presentations given in class 30%
- Written project report on final project 10%
- Final project presentation and demo 60%

Student Projects

- Many possibilities
  - Focus on hardware (e.g., measure bus or instruction times)
  - Design and implement traffic management / policing
  - Find the limits of a particular NP
  - Compare NP architectures
  - Implement application-layer functionality
  - Other (e.g., measure the effect of security on speed)
- Students choose their topic (subject to approval)

Project Teams

- Possibilities
  - Work alone
  - Work in a group of two
  - Work in a group of three or more
- Students choose group composition and size
- Note: project only approved if commensurate with group size

Project Administration

- Register group (week 2)
- Submit a topic for approval (week 4)
- Give a preliminary proposal (week 5)
- Report on status (week 9–10)
- Demonstrate project and turn in report (weeks 13–15)

Example Student Projects

- Network Address Translator (NAT box)
- Web load balancer
- IPsec implementation
- Configurable Internet firewall
- Traffic monitor (collect per-flow statistics)
- Virtual Private Network router
- Intrusion detection system
- TCP terminator
- IPv6 forwarder

Recommended Textbook

Comer, D., *Network Systems Design Using Network Processors*, Intel IXP Version, Prentice Hall, 2004. ISBN 0-13-141792-4.

Student Reaction

- Enthusiastic response
  - ‘‘Excellent course’’
  - ‘‘One of the most interesting and fun courses I have taken at Purdue’’
- Projects that go beyond the minimum
- High course evaluations

Questions?

PART III

An Example Undergraduate Course On Network Processors

Goals

- Become familiar with concept of network processor
- Appreciate that the field is new and evolving
- Gain experience programming one network processor
- Implement basic protocol processing
  - Layer 2 bridging
  - Packet header parsing
- Be able to characterize and describe features of network processors

Organization

- Lecture course plus lab
- Professor
  - Lectures throughout semester
  - Covers concepts and big picture
- Students
  - Learn from a textbook and lectures
  - Program in lab sections under supervision of TA
  - Begin with simplified API
  - Write small pieces of code

Topics

- Network systems and protocol processing
- History of network systems implementation
- Classification and classification languages
- Switching fabric concepts
- Motivation for network processors
- Survey of network processor architectures

Topics
(continued)

- Detailed example of one network processor
  - Architecture of each piece
  - I/O and internal memory interfaces
  - Programming model and structure of software
  - Cross-development environment
  - Examples of code

What Students Do Not Learn

- Engineering details
- How to create large, complex network systems
- All possible optimizations
- All architectural details
- How to make design tradeoffs
- Products and associated economic costs

Example Grading Scheme

- In-class quizzes 5%
- Midterm and final exams 35%
- Programming projects in lab 60%

The Lab Scheduling Problem

- Undergrads have little background
- First half of course covers general material
  - Network systems
  - Alternative implementations
  - Architectures
- Second half of course presents details of one NP
  - Hardware
  - Programming model
- Question: what lab projects can students do during the first half of the course?

Our Solution

- Present students with a higher-level programming system
  - Provide a simplified API that hides details
  - Make it easy to transmit or receive packets
- Have students use embedded processor
  - Bridge
  - IP fragmenter
- Defer microengine programming to second half of course

Simplified API

- Handles all I/O details
- Supports protocols that use retransmission
- Introduces students to asynchronous input
- Functions

Function	Purpose
onstartup()	Called once at initialization
onshutdown()	Called once at termination
newfbuf()	Allocate a frame buffer
recvframe()	Called when frame arrives
sendframe()	Used to transmit an outgoing frame
periodic_call()	Start a periodic timer
delayed_call()	Invoke a function after a delay
cancel_call()	Cancel a timer

Student Lab Projects (Intel)

- Using simplified API on embedded processor
  - Compile, download, and run code
  - Packet analyzer (IP/TCP/ARP)
  - Layer 2 bridge
  - IP fragmenter
  - Traffic classifier (i.e., packet analyzer)

Student Lab Projects (Intel) (continued)

- Using microcode on packet engine
  - Compile, download, and run a program
  - Classifier microblock
  - Frame forwarding microblock, where destination depends on classification

Textbooks

- Main text

Comer, D., *Network Systems Design Using Network Processors*, Intel IXP Version, Prentice Hall, 2004. ISBN 0-13-141792-4.

- Lab Manual

Comer, D., *Hands-On Networking With Internet Applications*, 2nd Edition, Prentice Hall, 2004.

Student Reaction

- Enthusiastic response
  - ‘‘Awesome class’’
  - ‘‘Best class I have ever taken at Purdue’’
- Enjoyed working with real hardware
- High course evaluations

Questions?

PART IV

Laboratory Facilities For Hands-on Work With Network Processors

Why A Laboratory?

- Absolutely essential: students learn by doing
- Reinforces concepts presented in class
- Exposes students to new (unusual) hardware
- Gives students concrete understanding of details
- Keeps courses tied to reality

Equipment Needed For A Lab

- Two types
- Front-end facilities
  - Conventional workstations
  - Connected to production network
  - Run standard OS
  - Used to prepare software
- Back-end facilities
  - Used for experimentation
  - Students download/configure

Comer’s Xinu Lab

- Established in 1984
- Provides hands-on access to hardware
- Used for research and education
  - Operating systems
  - Networking and Internetworking

Facilities In The Xinu Lab†

- Front-end systems
  - 24 workstations running Linux
  - Connected via gigabit Ethernet
- Back-end systems
  - 85 PCs
  - Miscellaneous routers, load balancer, etc.
- Networks
  - Gigabit Ethernet (production)
  - Various 10/100/1000 Ethernets (experimental)

†Thanks to: Intel, IBM, Cisco, Agere, AT&T, and others.

Front-end Systems In Comer's Lab

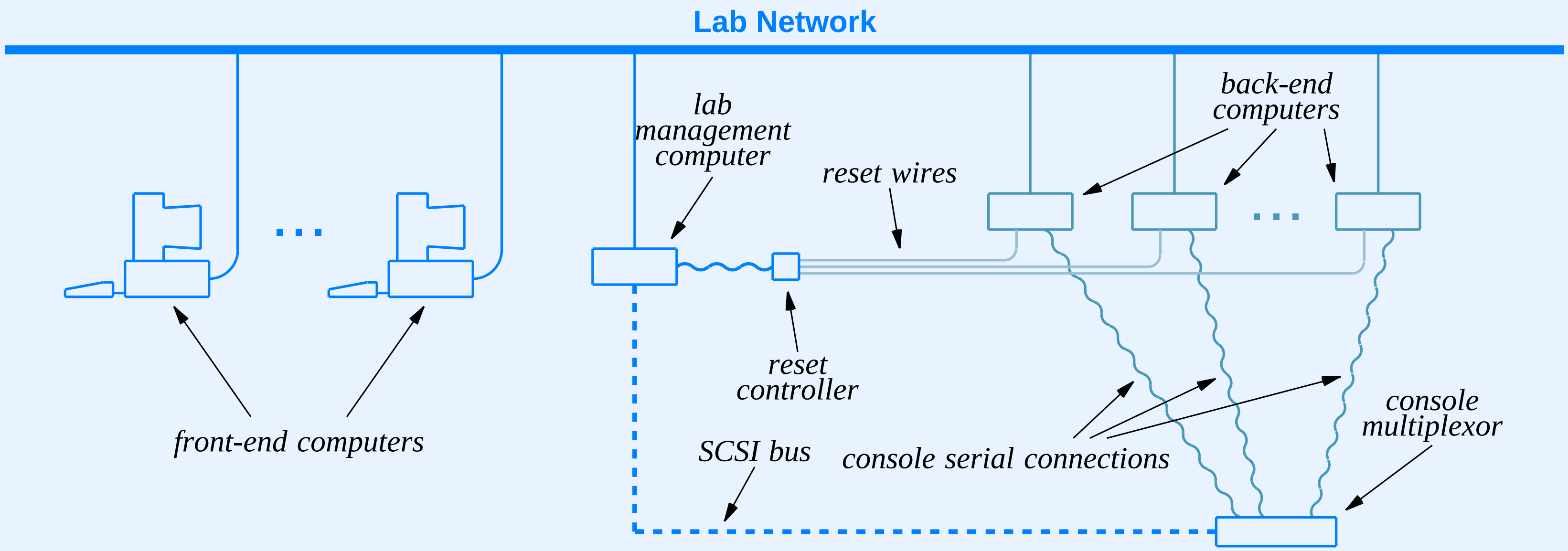
Normal View Of Xinu Lab

Back-end Systems In Comer's Lab

Lab Infrastructure

- Allows remote access to all facilities
- Software designed and built at Purdue
- Provides
  - Automated allocation of back-ends
  - Image download
  - Communication with back-end console
  - Control (reboot)

Conceptual Interconnections



- Commercial terminal multiplexor
- Custom reboot hardware

Automated Reconfigurable Testbed System (ART)

- Introduced in 2002
- Uses VLAN switch
- Provides automated connection of back-ends to networks
- Allows user to define and store configuration
- Offers GUI interface

Network Processors In The Xinu Lab

- Added in 2001
- Currently have
  - 22 Intel IXP1200 systems → 2400s
  - 2 Agere Payload Plus 2.5 systems → APP550s
  - 2 IBM NP4GS3 systems → ???
- Other lab equipment used with network processors
  - Hubs
  - Switches
  - Cables

Reference Platform

- Provided by vendor
- Targeted at potential customers
- Usually includes
  - Hardware testbed
  - Simulator / emulator
  - Cross-development software
  - Download and bootstrap software
  - Reference implementations

Two Types Of Reference Platforms

- Stand-alone
  - Separate chassis and power supply
  - Contains control processor plus NP system
- Single-board testbed
  - Plugs into control system (usually a PC)
  - Controlled via bus

Example Reference Hardware (Intel)

- Single-board network processor testbed
- Plugs into PCI bus on a PC
- Code name *Bridal Veil*
- Manufactured by Radisys

Items On The Intel Bridal Veil Reference System

Quantity or Size	Item
1	IXP1200 network processor (232MHz)
8	Mbytes of SRAM memory
256	Mbytes of SDRAM memory
8	Mbytes of Flash ROM memory
4	10/100 Ethernet ports
1	Serial interface (console)
1	PCI bus interface
1	PMC expansion site

Intel Reference Software

- Known as *Software Development Kit (SDK)*
- Runs on PC
- Includes:

Software	Purpose
C compiler	Compile programs for the StrongARM
MicroC compiler	Compile programs for the microengines
Assembler	Assemble programs for the microengines
Downloader	Load software into the network processor
Monitor	Communicate with the network processor and interact with running software
Bootstrap	Start the network processor running
Reference Code	Example programs for the IXP1200 that show how to implement basic functions

External Access

- SDRAM accessed via SDRAM bus
- SRAM and Flash accessed via SRAM bus
- Ethernet ports accessed via IX bus
- Code and data downloaded via PCI bus
- NFS accessed via PCI bus
- StrongARM accessed via
  – Serial line (console)
  – Telnet

Basic Paradigm

- Build software on conventional computer
- Load into reference system
- Test / measure results

Our Requirements

- Intel SDK designed to use
  - Windows system (compile)
  - Unix downloader
- Our requirement
  - No Windows
- Solution
  - Windows emulator when needed (Wine)

Questions?


STOP