



The Flexible Open-Source Networking Platform

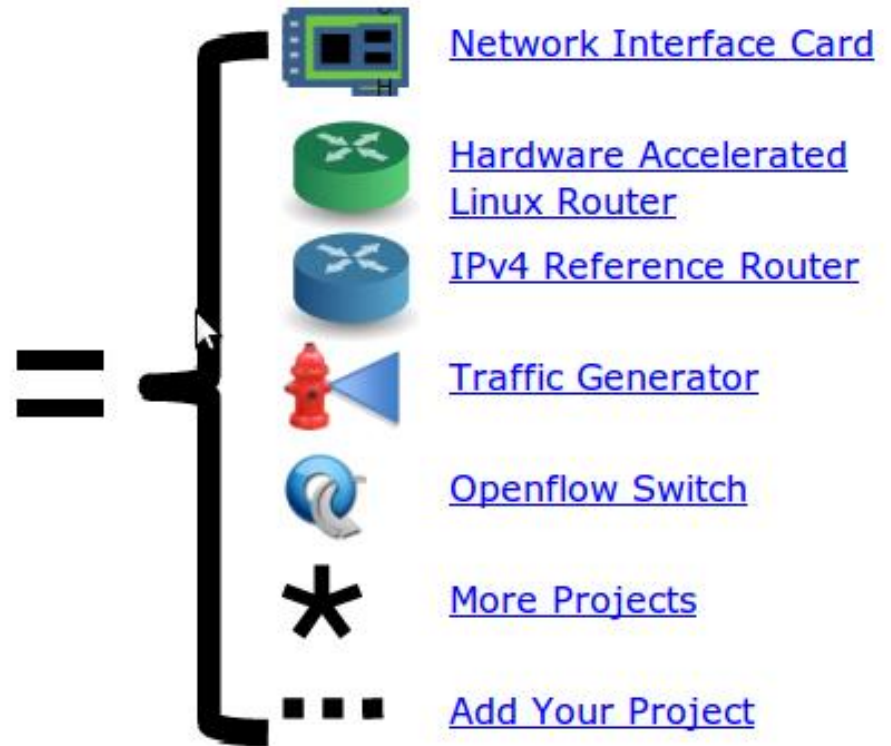
Noa Zilberman
University of Cambridge

- I. Overview**
- II. Hardware overview**
- III. Research projects**
- IV. Teaching**
- V. Community and Events**
- VI. Conclusion**

Section I: Overview

NetFPGA = Networked FPGA

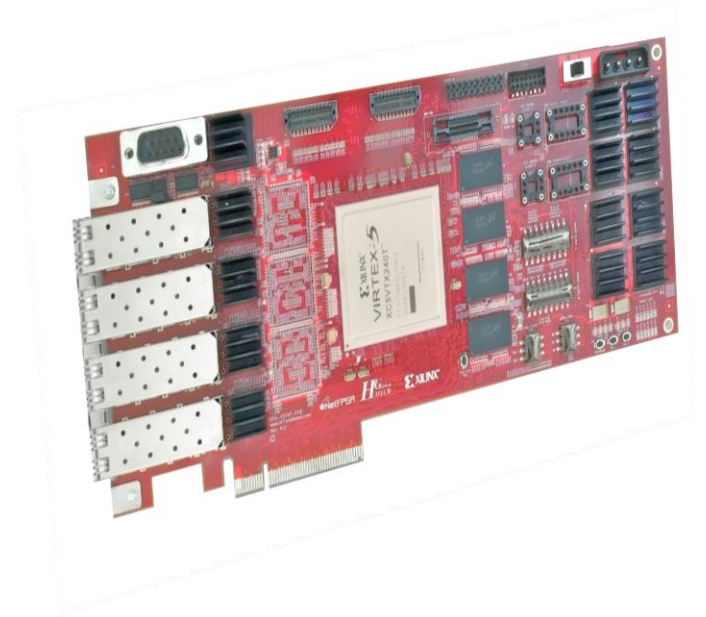
A line-rate, flexible, open networking platform for teaching and research



NetFPGA consists of...

Four elements:

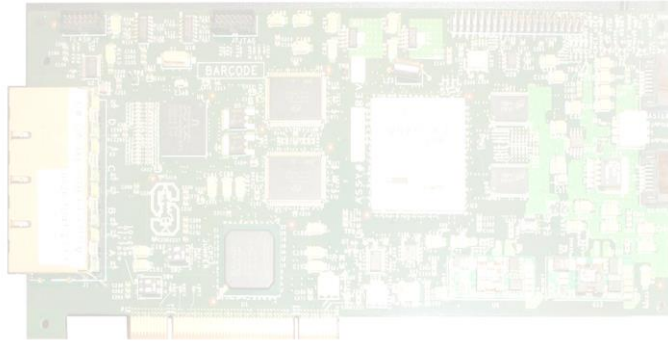
- **NetFPGA board**
- **Tools + reference designs**
- **Contributed projects**
- **Community**



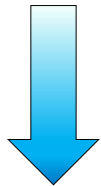
NetFPGA GitHub Organization

 The Interwebs  <http://www.netfpga.org>

NetFPGA Family of Boards



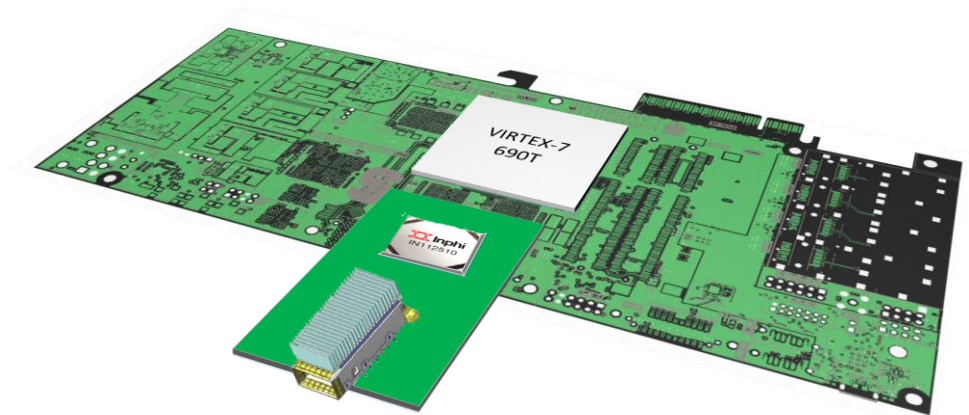
NetFPGA-1G (2006)



NetFPGA-10G (2010)



NetFPGA-1G-CML (2014)

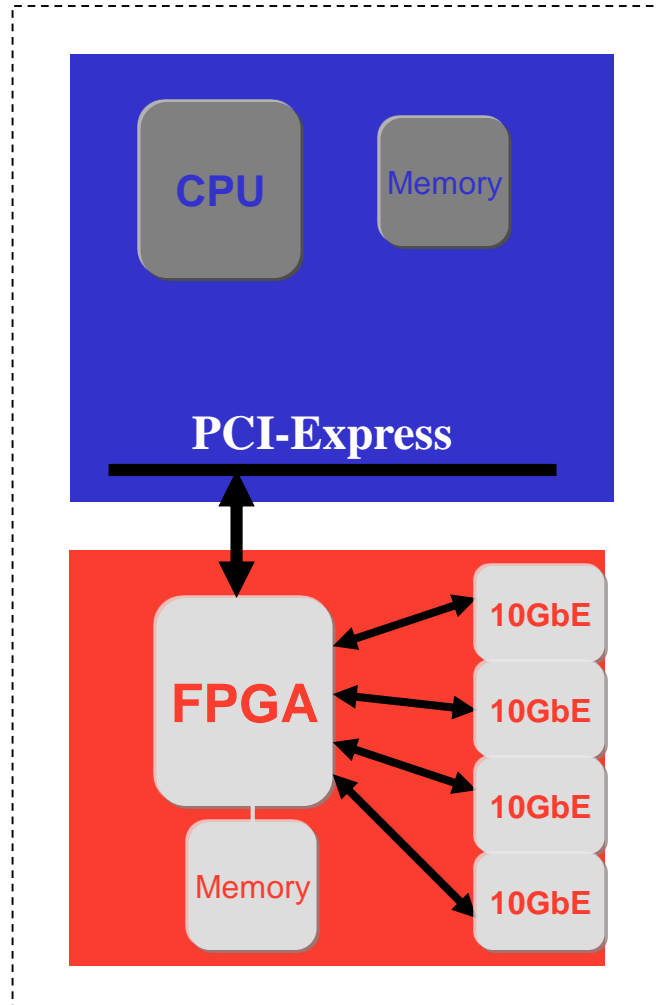


NetFPGA SUME (2014)

NetFPGA board

Networking Software running on a standard PC

A hardware accelerator built with Field Programmable Gate Array driving 1/10/100Gb/s network links



PC with NetFPGA



Tools + Reference Designs

Tools:

- **Compile designs**
- **Verify designs**
- **Interact with hardware**

Reference designs:

- **Router (HW)**
- **Switch (HW)**
- **Network Interface Card (HW)**
- **Router Kit (SW)**
- **SCONE (SW)**

Community

Wiki

- **Documentation**
 - User's Guide *“so you just got your first NetFPGA”*
 - Developer's Guide *“so you want to build a ...”*
- **Encourage users to contribute**

Forums

- **Support by users for users**
- **Active community - 10s-100s of posts/week**

International Community

Over 1,000 users, using 3,115 cards at
150 universities in 40 countries



NetFPGA's Defining Characteristics

- **Line-Rate**

- Processes back-to-back packets
 - Without dropping packets
 - At full rate
- Operating on packet headers
 - For switching, routing, and firewall rules
- And packet payloads
 - For content processing and intrusion prevention

- **Open-source Hardware**

- Similar to open-source software
 - Full source code available
 - BSD-Style License for 1G and LGPL 2.1 for 10G
- But harder, because
 - Hardware modules must meeting timing
 - Verilog & VHDL Components have more complex interfaces
 - Hardware designers need high confidence in specification of modules

Test-Driven Design

- **Regression tests**
 - Have repeatable results
 - Define the supported features
 - Provide clear expectation on functionality
- ***Example: Internet Router***
 - Drops packets with bad IP checksum
 - Performs Longest Prefix Matching on destination address
 - Forwards IPv4 packets of length 64-1500 bytes
 - Generates ICMP message for packets with TTL ≤ 1
 - Defines how to handle packets with IP options or non IPv4
 - ... and dozens more ...
 - Every feature is defined by a regression test***

Who, How, Why

Who uses the NetFPGA?

- Researchers
- Teachers
- Students

How do they use the NetFPGA?

- To run the Router Kit
- To build modular reference designs
 - IPv4 router
 - 4-port NIC
 - Ethernet switch, ...

Why do they use the NetFPGA?

- To measure performance of Internet systems
- To prototype new networking systems

Section II: Hardware Overview

Available Now

NetFPGA-1G-CML

- **FPGA Xilinx Kintex7**
- **4x 10/100/1000 Ports**
- **PCIe Gen.2 x4**
- **QDRII+-SRAM, 4.5MB**
- **DDR3, 512MB**
- **SD Card**
- **Expansion Slot**



Available Now

NetFPGA-10G

- **FPGA Xilinx Virtex5**
- **4 SFP+ Cages**
 - 10G Support
 - 1G Support
- **PCIe Gen.1 x8**
- **QDRII-SRAM, 27MB**
- **RLDRAM-II, 288MB**
- **Expansion Slot**

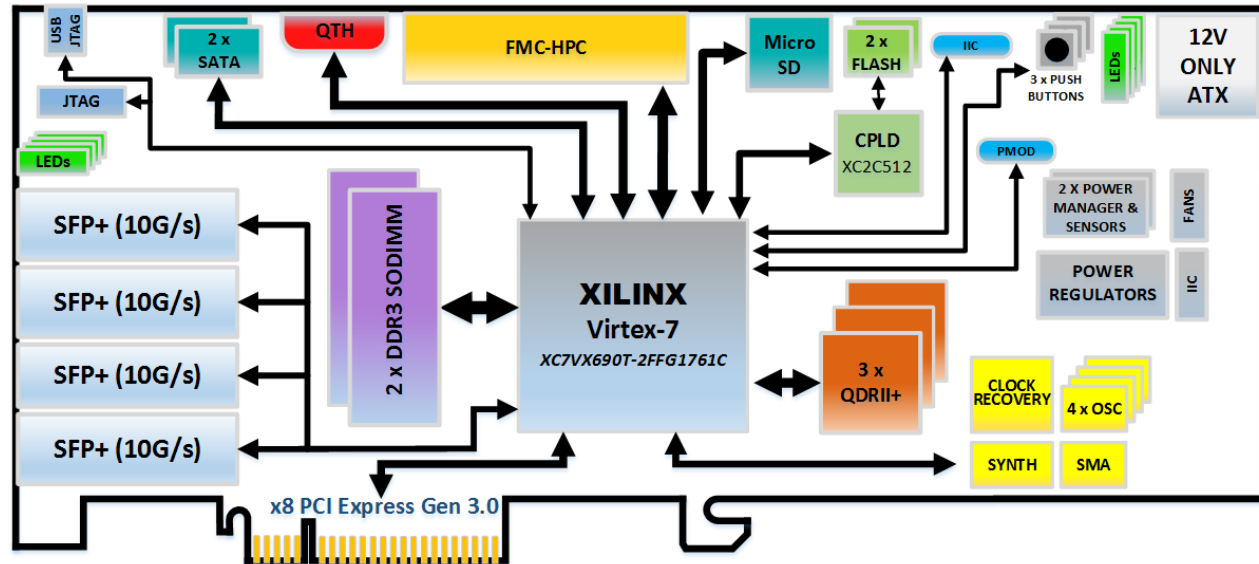


Available Q3/14

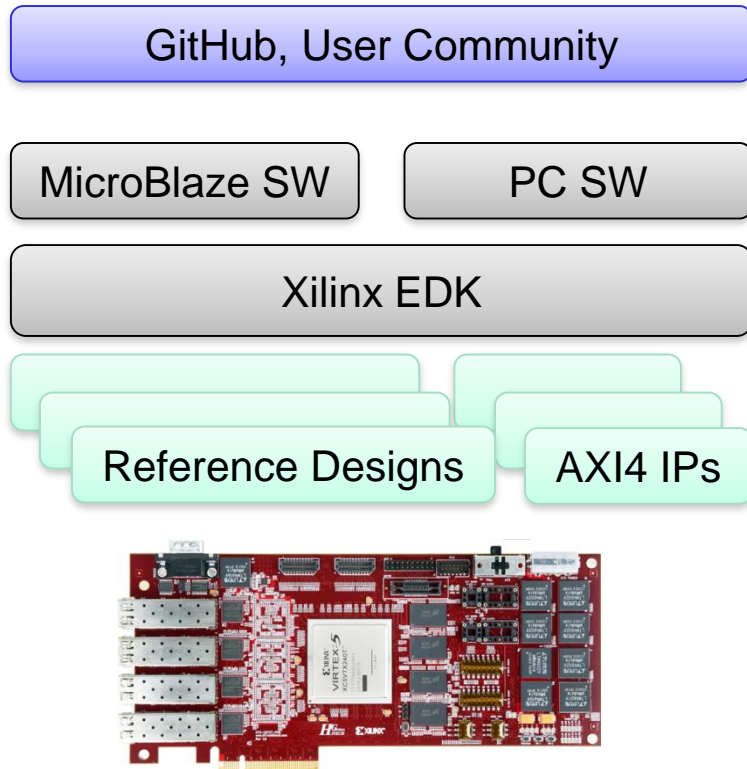
NetFPGA SUME

FPGA Xilinx Virtex7

- 4 SFP+ Cages
 - 10G Support
 - 1G Support
- 18x13.1Gb/s Additional Serial Links
- PCIe Gen.3 x8
- QDRII+-SRAM, 3x72Mb, 500MHz
- DDR3 SoDIMM, 2x4GB, 1866MT/s
- Expansion Slot
- Micro-SD



Beyond Hardware



- **NetFPGA Board**
- **Xilinx EDK based IDE**
- **Reference designs with ARM AXI4**
- **Software (embedded and PC)**
- **Public Repository**
- **Public Wiki**

Section III: Research Projects

OpenFlow

- **The most prominent NetFPGA success**
- **Has reignited the Software Defined Networking movement**
- **NetFPGA enabled OpenFlow**
 - A widely available open-source development platform
 - Capable of line-rate and
- **was, until its commercial uptake, the reference platform for OpenFlow.**

Contributed Projects

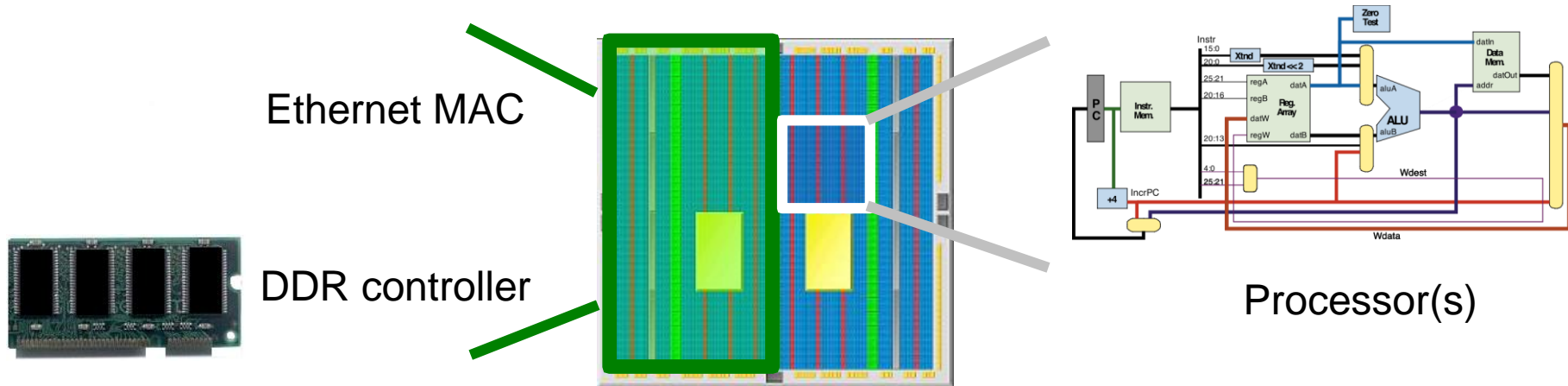
Platform	Project	Contributor
1G	OpenFlow switch	Stanford University
	Packet generator	Stanford University
	NetFlow Probe	Brno University
	NetThreads	University of Toronto
	zFilter (Sp)router	Ericsson
	Traffic Monitor	University of Catania
	DFA	UMass Lowell
10G	Bluespec switch	MIT/SRI International
	Traffic Monitor	University of Pisa
	NF1G legacy on NF10G	Uni Pisa & Uni Cambridge
	Simple/better DMA core	Stanford RAMcloud project

Some Ongoing Projects

- **Computing**
 - Stand alone computing unit (CHERI soft core)
 - Security and capabilities over NetFPGA-10G (Cambridge & SRI)
- **Measurements**
 - Open Source Network Tester (6 contrib groups)
 - Accurate Internet measurements (Cambridge & TAU)
- **SDN**
 - OpenFlow switch 1.4 (Cambridge & SRI)



Soft Processors in FPGAs



- Soft processors: processors in the FPGA fabric
- User uploads program to soft processor
- Easier to program software than hardware in the FPGA
- Could be customized at the instruction level

Open Source Network Tester

Long development cycles and high cost create a requirement for open-source network testing

- **Open-source hardware platform**
- **For research and teaching community**



- high-performance (40GbE support)
- low-cost (\$1600, cost of NF board)
- flexible
- scalable
- open-source community

www.osnt.org

OSNT Use Cases

OSNT flexibility provides support for a wide range of use-cases

- **OSNT-TG (Traffic Generator)**
 - A single card, generating packets on four 10GbE ports
- **OSNT-MON (Traffic Monitor)**
 - a single card, capturing packets from four 10GbE ports
- **Hybrid OSNT**
 - the combination of OSNT-TG and OSNT-MON
 - On a single card
- **Scalable OSNT**
 - Coordinating multiple generators and monitors
 - Synchronized by a common time-base

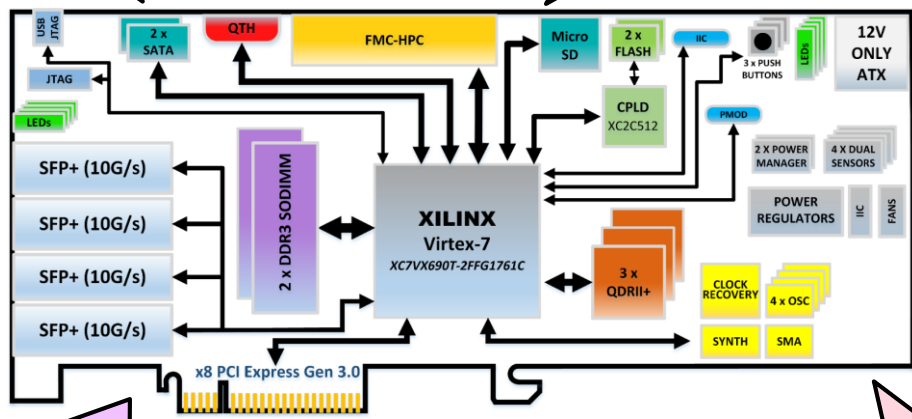
Future

NetFPGA SUME A Technology Enabler

Stand Alone Device

100Gb/s Switch

PHY & MAC



PCIe Host Interface

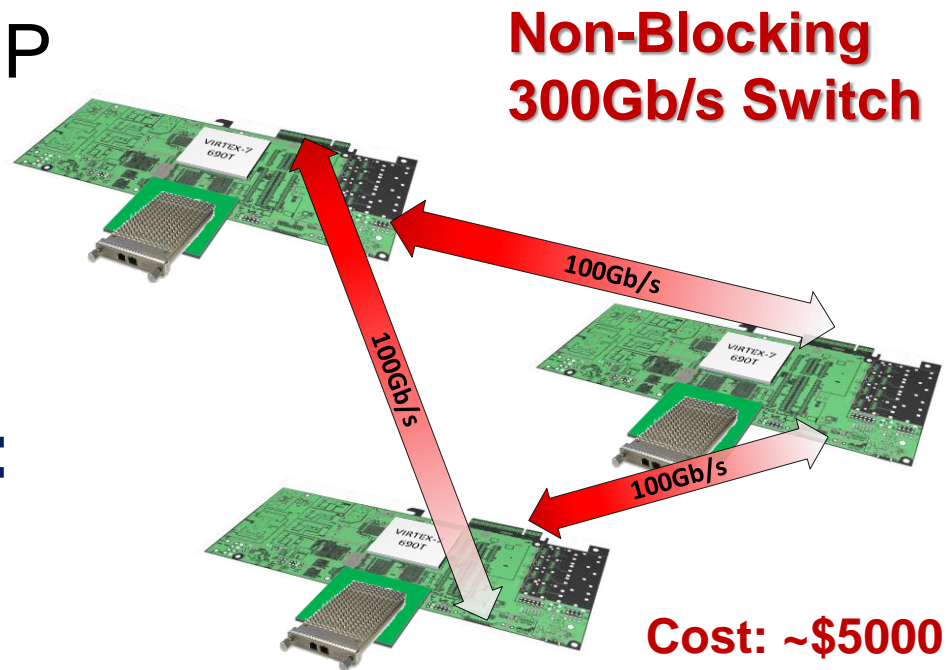
Interconnect

Future

100Gb/s Aggregation

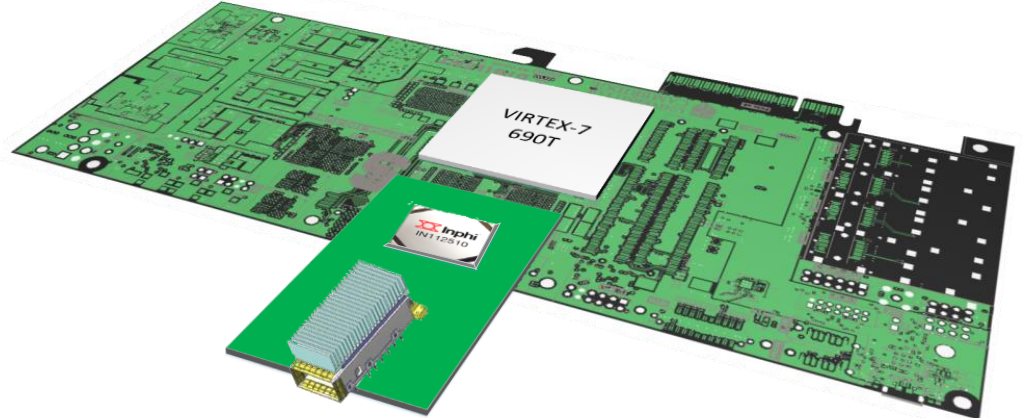
- Need a development platform that can aggregate 100Gb/s for:
 - Operating systems
 - Protocols beyond TCP

- **NetFPGA SUME** can:
 - Aggregate 100Gb/s as Host Bus Adapter
 - Be used to create large scale switches



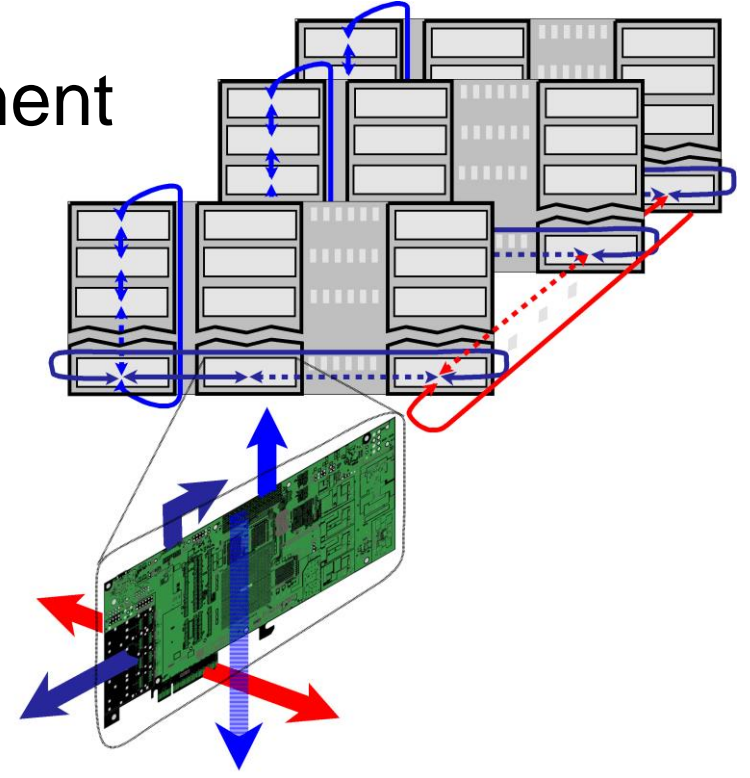
Power Efficient MAC

- Need for 100Gb/s power-saving MAC design (e.g. lights-out MAC)
- Porting MAC design to SUME permits:
 - Power measurements
 - Testing protocol's response
 - Reconsideration of power-saving mechanisms
 - Evaluating suitability for complex architectures and systems



Interconnect

- **Novel Architectures with line-rate performance**
 - A lot of networking equipment
 - Extremely complex
- **NetFPGA SUME allows prototyping a *complete* solution**



Camcube
N x N xN Hyper-cube

How might we use NetFPGA?

Well, I'm not sure about you but here is a list I created:

- Build an accurate, fast, line-rate NetDummy/nistnet element
- A flexible home-grown monitoring card
- Evaluate new packet classifiers
 - (and application classifiers, and other neat network apps....)
- Prototype a full line-rate next-generation Ethernet-type
- Trying any of Jon Crowcrofts' ideas (Sourceless IP routing for example)
- Demonstrate the wonders of Metarouting in a different implementation (dedicated hardware)
- Provable hardware (using a C# implementation and kiwi with NetFPGA as target)
- Hardware supporting Virtual Routers
- Check that some brave new idea actually works
 - e.g. Rate Control Protocol (RCP), Multipath TCP
- A flexible home-grown monitoring card
- MOOSE implementation
- IP address anonymization
- Evaluate new packet classifiers
- SSL decoding "bump in the wire"
- Xen specialist (and application classifiers, and other neat network apps....)
- computational co-processor
- Distributed computational co-processor
- Prototype a full line-rate next-generation Ethernet-type
- IPv6 - IPv4 gateway (6in4, 4in6, 6over4, 4over6,)
- Netflow v9 reference
- PSAMP reference
- Trying any of Jon Crowcrofts' ideas (Sourceless IP routing for example)
- IPFIX reference
- Different driver/buffer interfaces (e.g. PFRING)
- or "escalators" (from gridprobe) for faster network monitors
- Firewall reference
- GPS packet-injection things
- High-Speed Host Bus Adapter reference implementations
 - Infiniband
 - iSCSI
 - Myranet
 - h/w channel
- Smart Disk adapter (presuming a direct-disk interface)
- Software Defined Radio (SDR) directly on the FPGA (probably UWB only)
- Routing accelerator
 - Hardware route-reflector
 - Internet exchange route accelerator
- Hardware channel bonding reference implementation
- TCP sanitizer
- Other protocol sanitizer (applications... UDP DCCP, etc.)
- Full and complete Crypto NIC
- IPSec endpoint/ VPN appliance
- VLAN reference implementation
- metarouting implementation
- Virtual switch something>
- intelligent proxy
- application embargo-er
- Layer-4 gateway
- h/w gateway for VoIP/SIP/skype
- h/w gateway for video conference spaces
- security pattern/rules matching
- Anti-spoof traceback implementations (e.g. BBN stuff)
- IPTv multicast controller
- Intelligent IP-enabled device controller (e.g. IP cameras or IP powerm...)
- DES breaker
- platform for flexible NIC API evaluations
- snmp statistics reference implementation
- sflow (hp) reference implementation
- packet sampling reference implementation)
- implementation of zeroconf/netconf configuration language for route...
- h/w openflow and (simple) NOX controller in one...
- Network Flow Processor (NFP) (and...)
- inline compression
- hardware accelerator for TOR
- load-balancer
- openflow with (netflow, ACL,)
- reference NAT device
- active measurement kit
- network discovery tool
- passive performance measurement
- active sender control (e.g. performance feedback fed to endpoints for...
- Prototype platform for NON-Ethernet or near-Ethernet MACs
 - Optical LAN (no buffers)

How might YOU use NetFPGA?

- Build an accurate, fast, line-rate NetDummy/nistnet element
- A flexible home-grown monitoring card
- Evaluate new packet classifiers
 - (and application classifiers, and other neat network apps....)
- Prototype a full line-rate next-generation Ethernet-type
- Trying any of Jon Crowcroft's ideas (Sourceless IP routing for example)
- Demonstrate the wonders of Metarouting in a different implementation (dedicated hardware)
- Provable hardware (using a C# implementation and kiwi with NetFPGA as target h/w)
- Hardware supporting Virtual Routers
- Check that some brave new idea actually works
 - e.g. Rate Control Protocol (RCP), Multipath TCP,
- toolkit for hardware hashing
- MOOSE implementation
- IP address anonymization
- SSL decoding "bump in the wire"
- Xen specialist nic
- computational co-processor
- Distributed computational co-processor
- IPv6 anything
- IPv6 – IPv4 gateway (6in4, 4in6, 6over4, 4over6,)
- Netflow v9 reference
- PSAMP reference
- IPFIX reference
- Different driver/buffer interfaces (e.g. PFRING)
- or "escalators" (from gridprobe) for faster network monitors
- Firewall reference
- GPS packet-timestamp things
- High-Speed Host Bus Adapter reference implementations
 - Infiniband
 - iSCSI
 - Myranet
 - Fiber Channel
- Smart Disk adapter (presuming a direct-disk interface)
- Software Defined Radio (SDR) directly on the FPGA (probably UWB only)
- Routing accelerator
 - Hardware route-reflector
 - Internet exchange route accelerator
- Hardware channel bonding reference implementation
- TCP sanitizer
- Other protocol sanitizer (applications... UDP DCCP, etc.)
- Full and complete Crypto NIC
- IPSec endpoint/ VPN appliance
- VLAN reference implementation
- metarouting implementation
- virtual <pick-something>
- intelligent proxy
- application embargo-er
- Layer-4 gateway
- h/w gateway for VoIP/SIP/skype
- h/w gateway for video conference spaces
- security pattern/rules matching
- Anti-spoof traceback implementations (e.g. BBN stuff)
- IPTv multicast controller
- Intelligent IP-enabled device controller (e.g. IP cameras or IP powerm...)
- DES breaker
- platform for flexible NIC API evaluations
- snmp statistics reference implementation
- sflow (hp) reference implementation
- trajectory sampling (reference implementation)
- implementation of zeroconf/netconf configuration language for route...
- h/w openflow and (simple) NOX controller in one...
- Network RAID (multicast TCP with redundancy)
- inline compression
- hardware accelerator for TOR
- load-balancer
- openflow with (netflow, ACL,)
- reference NAT device
- active measurement kit
- network discovery tool
- passive performance measurement
- active sender control (e.g. performance feedback fed to endpoints for...
- Prototype platform for NON-Ethernet or near-Ethernet MACs
 - Optical LAN (no buffers)

Section IV: Teaching

NetFPGA in the Classroom

•Stanford University

- EE109 “Build an Ethernet Switch”
Undergraduate course for all EE students
<http://www.stanford.edu/class/ee109/>
- CS344 “Building an Internet Router” (since ‘05)
Quarter-long course targeted at graduates
<http://cs344.stanford.edu>

•Rice University

- Network Systems Architecture (since ‘08)
<http://comp519.cs.rice.edu/>

•Cambridge University

- Build an Internet Router (since ‘09)
Quarter-long course targeted at graduates
<http://www.cl.cam.ac.uk/teaching/current/P33/>

•University of Wisconsin

- CS838 “Rethinking the Internet Architecture”
<http://pages.cs.wisc.edu/~akella/CS838/F09/>

•University of Bonn

- “Building a Hardware Router”
<http://bit.ly/Kmo0rA>

See: <http://netfpga.org/teachers.html>

Components of NetFPGA Course

- **Documentation**
 - System Design
 - Implementation Plan
- **Deliverables**
 - Hardware Circuits
 - System Software
 - Milestones
- **Testing**
 - Proof of Correctness
 - Integrated Testing
 - Interoperability
- **Post Mortem**
 - Lessons Learned

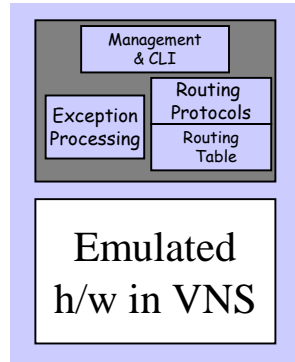
NetFPGA in the Classroom

- **Stanford CS344: “Build an Internet Router”**
 - Courseware available on-line
 - Students work in teams of three
 - 1-2 software
 - 1-2 hardware
 - Design and implement router in 8 weeks
 - Write software for CLI and PW-OSPF
 - Show interoperability with other groups
 - Add new features in remaining two weeks
 - Firewall, NAT, DRR, Packet capture, Data generator, ...

CS344 Milestones

1

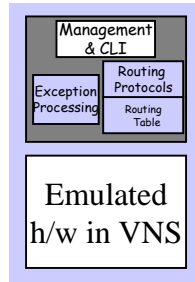
Build basic router



software
hardware

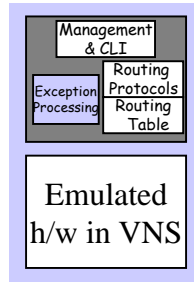
2

Command Line Routing Protocol Interface (PWOSPF)



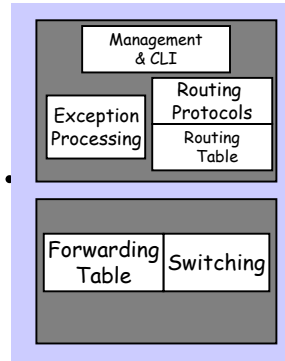
3

Routing Protocol (PWOSPF)



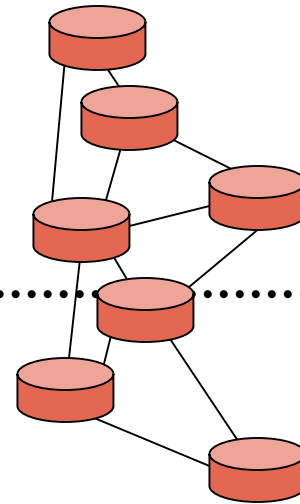
4

Integrate with H/W



5

Interoperability



6

Final Project

- Innovate and add!
- Presentations
- Judges

Learning Environment
Modular design
Testing

4-port non-learning switch

4-port learning switch

IPv4 router forwarding path

Integrate with S/W

Interoperability

Typical NetFPGA Course Plan

Week	Software	Hardware	Deliver
1	Verify Software Tools	Verify CAD Tools	Write Design Document
2	Build Software Router	Build Non-Learning Switch	Run Software Router
3	Cmd. Line Interface	Build Learning Switch	Run Basic Switch
4	Router Protocols	Output Queues	Run Learning Switch
5	Implement Protocol	Forwarding Path	Interface SW & HW
6	Control Hardware	Hardware Registers	HW/SW Test
7	Interoperate Software & Hardware		Router Submission
8	Plan New Advanced Feature		Project Design Plan
9	Show new Advanced Feature		Demonstration

Presentations



Stanford CS344

<http://cs344.stanford.edu>



Cambridge P33

<http://www.cl.cam.ac.uk/teaching/0910/P33/>

Section VI: Where Next?

To get started with your project

1. **New Software ideas? get familiar with the host-systems of the current reference (C and java)**
2. **replace them at will; no egos will be hurt**

OR

1. **New Hardware ideas? get familiar with hardware description language**
2. **Prepare for your project**
 - a) **Become familiar with the NetFPGA yourself**
 - b) **Go to a hands-on event**

Good practice is familiarity with hardware and software.... (and it isn't that scary - honest)

Scared by Verilog? Try our Online Verilog tutor (with NetFPGA extensions)

www-netfpga.cl.cam.ac.uk



Cambridge Verilog Tutor for NetFPGA



Cambridge Verilog Tutor for NetFPGA

Getting Started

Welcome to the Cambridge Verilog Tutor. This is a resource for students to learn the basics of Verilog.

Cambridge Users:

Login via Raven

External Users:

Login

Site information

For this website to function correctly, you are required to use a browser with cookies enabled, and which can create SSL connections.

In addition, Javascript and Flash are used to help teach more effectively, and enabling these technologies is advised, but not strictly required.

We believe that this website conforms to today's web standards. Any modern browser should be able to cope but there may be issues with older browsers.


Note:

For users with raven accounts, no registration is necessary. Just log in and get started.

Register or Reset account:

Email Address:

Register or Reset

Support for NetFPGA enhancements provided by 

Go to a hands-on camp



Stanford

Cambridge

Check out <http://www.netfpga.org/events.html>

Get a hands-on tutorial



Home Applications **Events** News Videos Wiki Forums About

[Upcoming Tutorials](#) [Camps](#) [Developer's Workshops](#) [Contest\(s\)](#)

Click [here](#) to view the events map.

Upcoming Tutorials [Top](#)

- [Technion](#)

- Date: April 08, 2014, 14:30
- Location: CS, Taub 337, Technion
- Goal: One hour NetFPGA Introductory
- Host: Noa Zilberman
- Website: <http://ceclub.technion.ac.il/upcoming.html#noazilberman14>
- Slides:

- [Tel-Aviv University](#)

- Date: April 07, 2014, 15:00
- Location: CS, Taub 337, Technion
- Goal: One hour NetFPGA Introductory
- Host: Noa Zilberman
- Website: http://www.eng.tau.ac.il/index.php?option=com_jevents&task=icalrepeat.detail&evid=1121&Itemid=292&year=2014&lang=eng-seminar-netfpga-the-flexible-open-source-networking-platform
- Slides:

NetFPGA website (www.netfpga.org)

Start with a board....

For US Universities (donations available)

- http://netfpga.org/donation_request.html

For Non-US Universities (donations available)

- <http://www.xilinx.com/member/xup/donation/request.htm>

For Non-Universities

- http://www.hitechglobal.com/Boards/PCIExpress_SFP+.htm
- <http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,400,1228&Prod=NETFPGA-1G-CML>

Acknowledgments (I)

NetFPGA Team at Stanford University (Past and Present):

Nick McKeown, Glen Gibb, Jad Naous, David Erickson,
G. Adam Covington, John W. Lockwood, Jianying Luo, Brandon Heller, Paul
Hartke, Neda Beheshti, Sara Bolouki, James Zeng,
Jonathan Ellithorpe, Sachidanandan Sambandan, Eric Lo

NetFPGA Team at University of Cambridge (Past and Present):

Andrew Moore, David Miller, Muhammad Shahbaz, Martin Zadnik
Matthew Grosvenor, Yury Audzevich, Neelakandan Manihatty-Bojan,
Georgina Kalogeridou, Jong Hun Han, Noa Zilberman, Gianni Antichi, Marco
Forconesi

All Community members (including but not limited to):

Paul Rodman, Kumar Sanghvi, Wojciech A. Koszek,
Yahsar Ganjali, Martin Labrecque, Jeff Shafer,
Eric Keller , Tatsuya Yabe, Bilal Anwer,
Yashar Ganjali, Martin Labrecque

Kees Vissers, Michaela Blott, Shep Siegel

Acknowledgements (II)



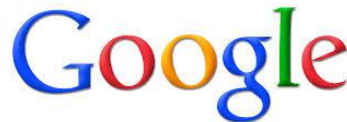
UNIVERSITY OF
CAMBRIDGE



EPSRC
Pioneering research
and skills



ALGO-LOGIC



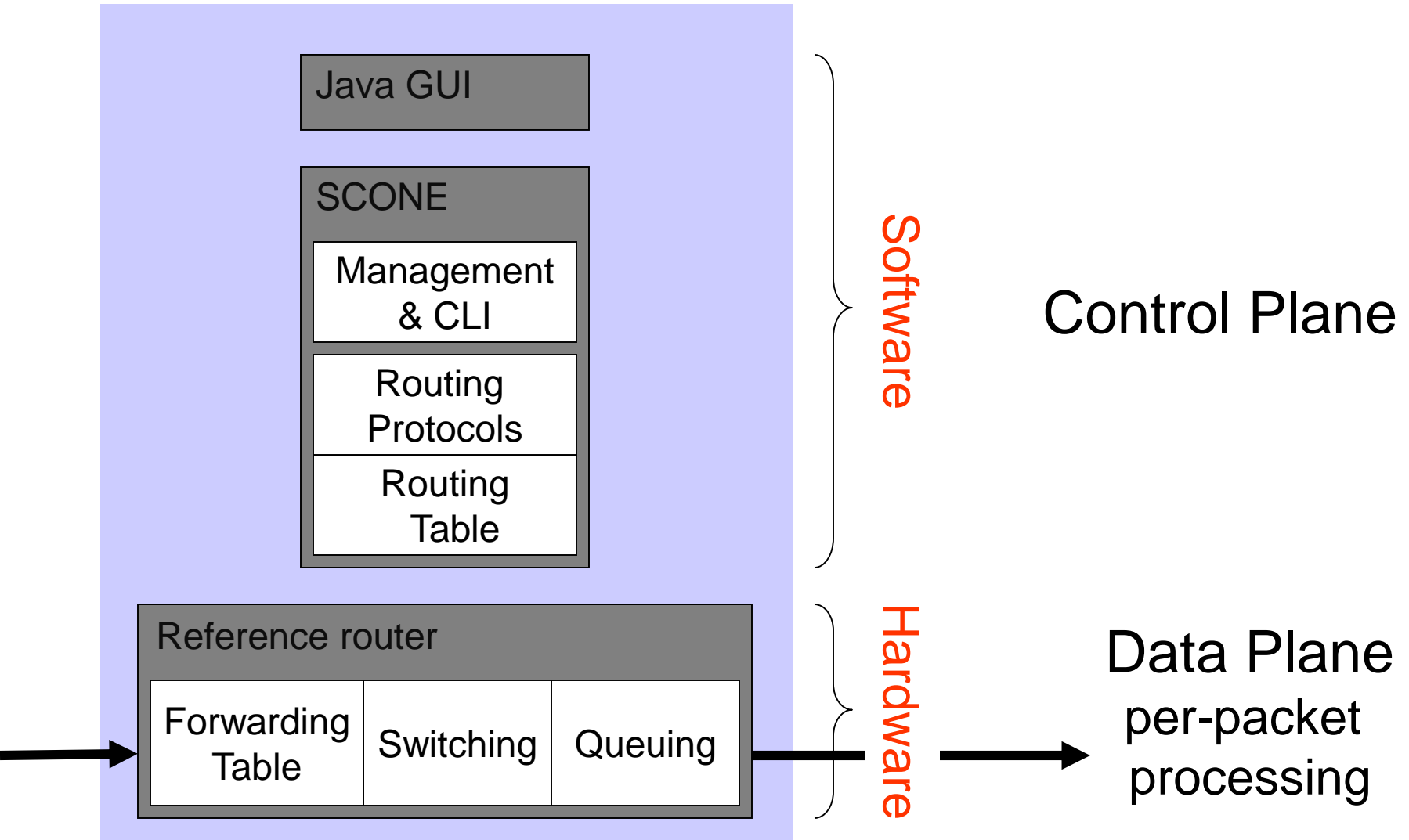
Disclaimer: Any opinions, findings, conclusions, or recommendations expressed in these materials do not necessarily reflect the views of the National Science Foundation or of any other sponsors supporting this project.

This effort is also sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL), under contract FA8750-11-C-0249. This material is approved for public release, distribution unlimited. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

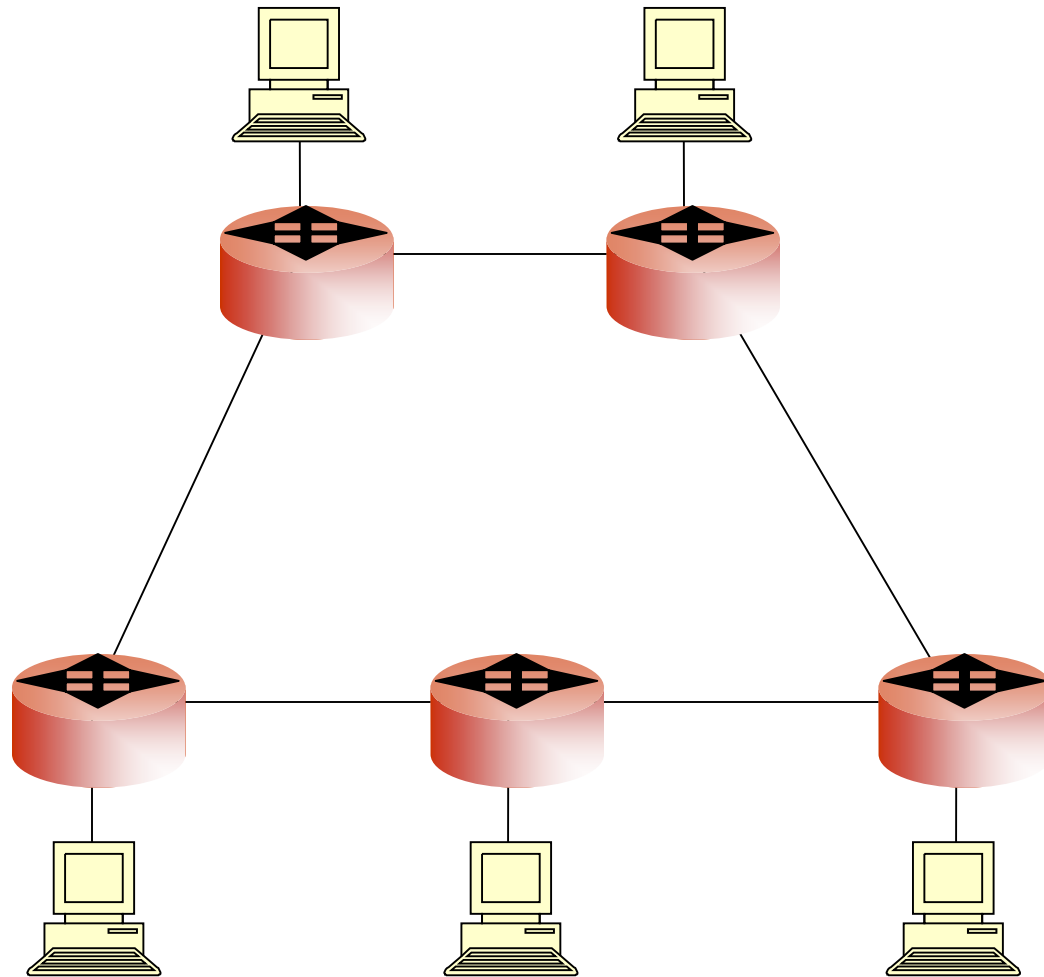
Thank You!

Appendix I: Example

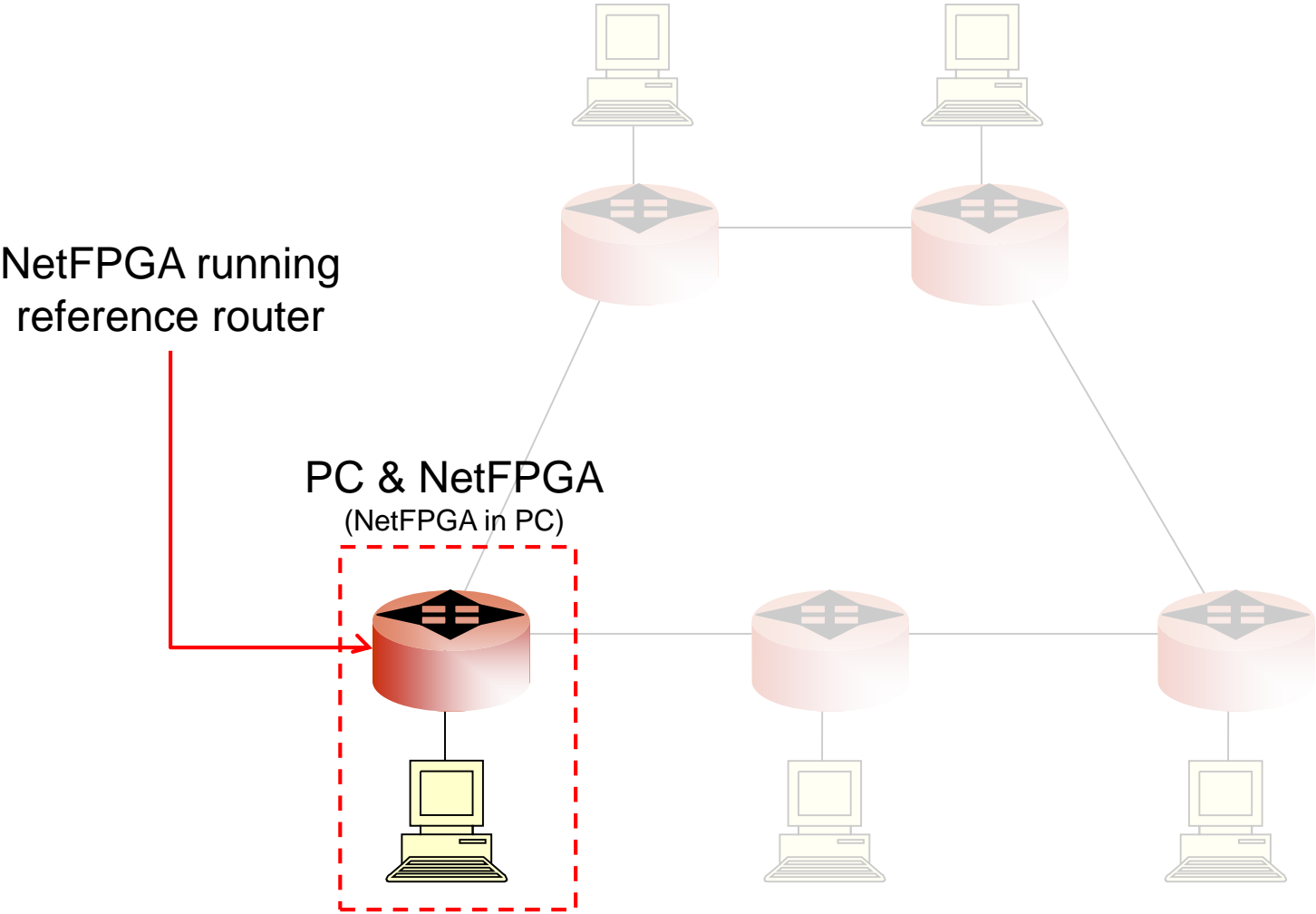
Operational IPv4 router



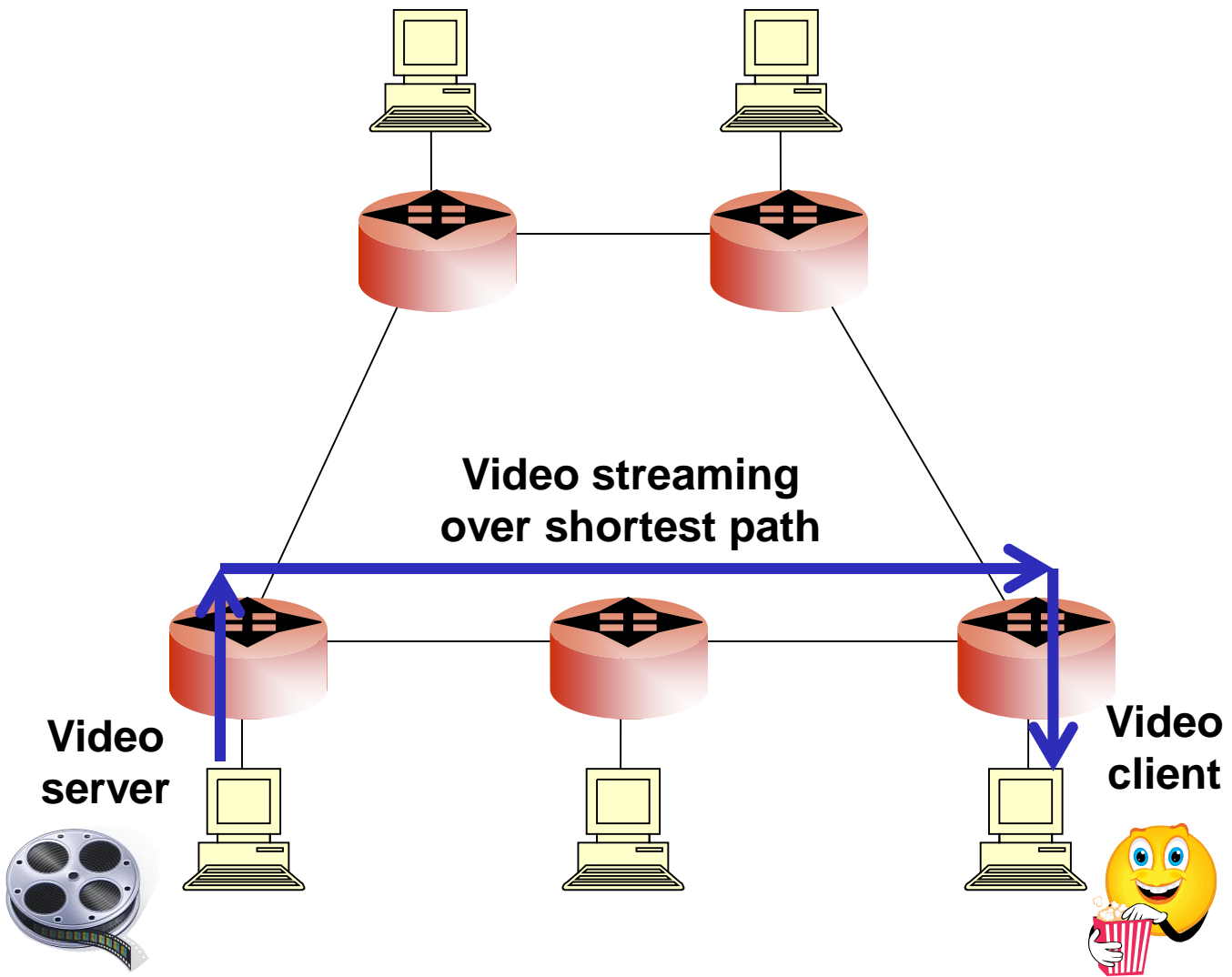
Streaming video



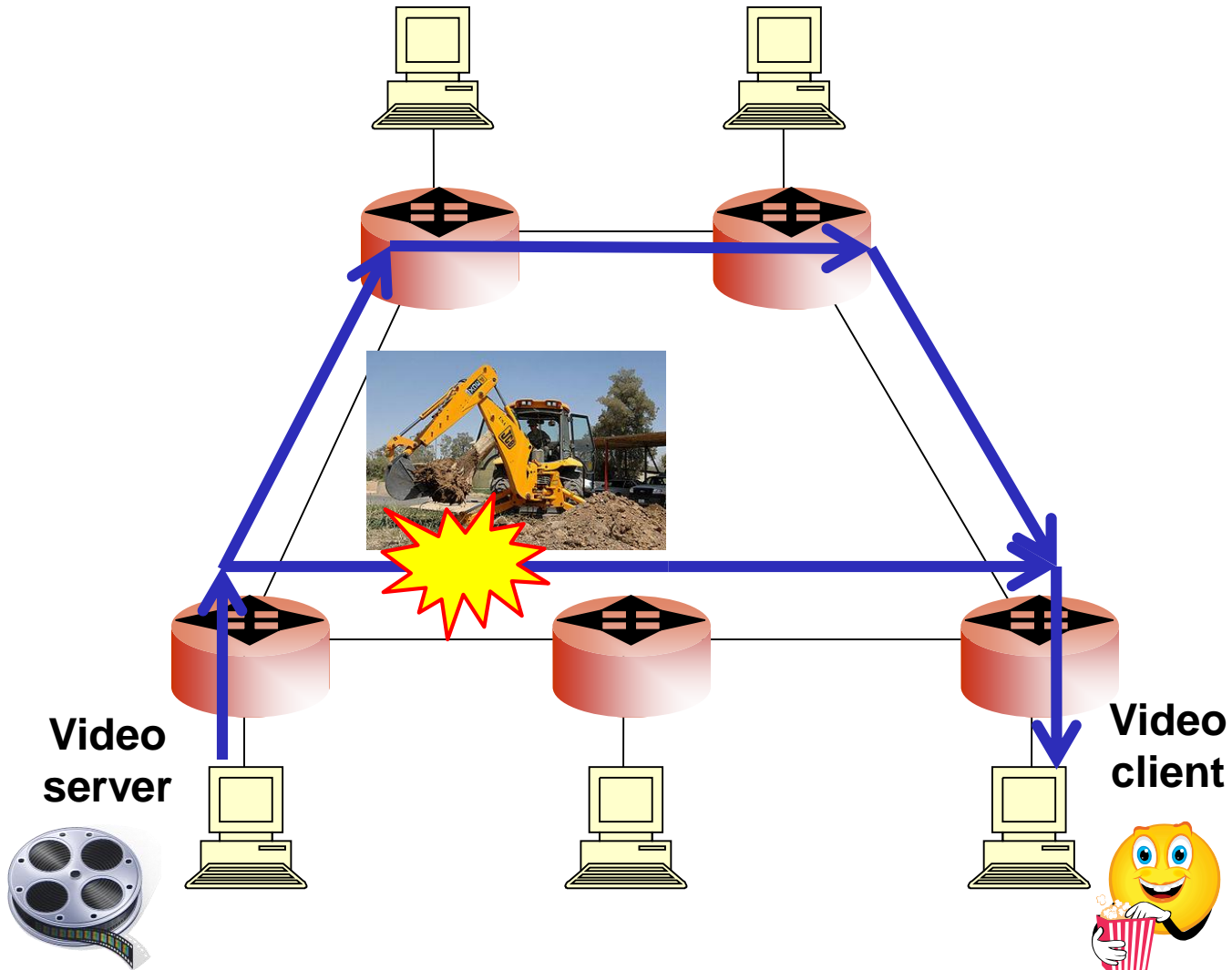
Streaming video



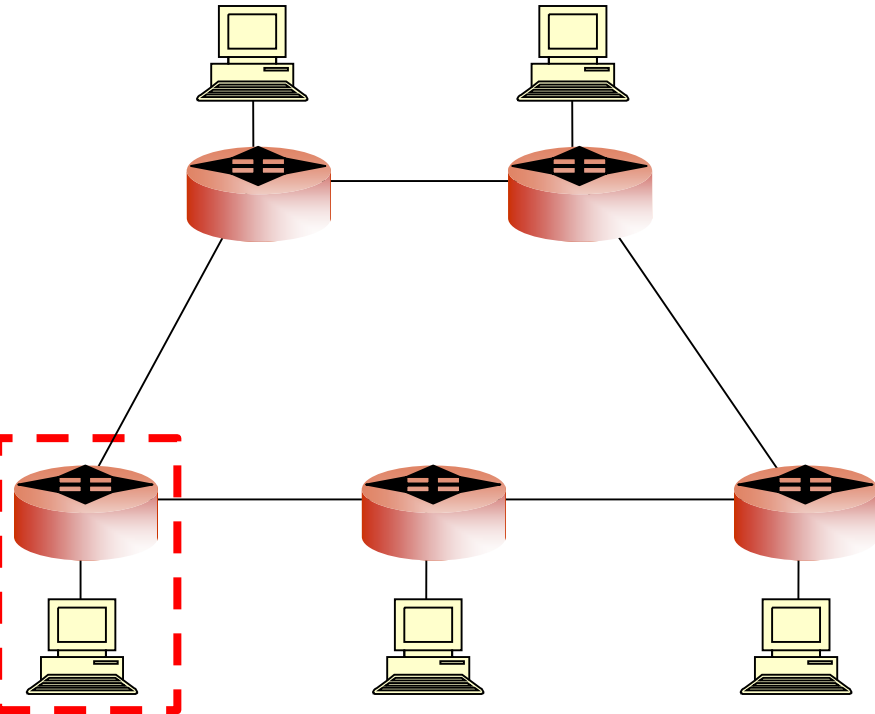
Streaming video



Streaming video



Observing the routing tables



Columns:

- Subnet address
- Subnet mask
- Next hop IP
- Output ports

Router Control Panel

Router Quickstart

Configuration Statistics Details

Router Configuration

Interface Configuration Load From File

Port Number	MAC Address	IP Address
0	00:00:00:00:01:01	192.168.3.1
1	100:00:00:00:01:02	192.168.2.2
2	200:00:00:00:01:03	192.168.1.2
3	300:00:00:00:01:04	192.168.15.2

Routing Table

Reset Entry

Modified	Index	Destination IP A...	Subnet Mask	NextHop IP A...	MAC0	CPU0	MAC1	CPU1	MAC2	CPU2	MAC3	CPU3
<input type="checkbox"/>	0	192.168.15.0	255.255.2...	0.0.0.0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	1	192.168.14.0	255.255.2...	192.168.3.2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	2	192.168.13.0	255.255.2...	192.168.3.2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	3	192.168.12.0	255.255.2...	192.168.3.2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	4	192.168.11.0	255.255.2...	192.168.3.2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	5	192.168.10.0	255.255.2...	192.168.3.2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	6	192.168.9.0	255.255.2...	192.168.3.2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	7	192.168.8.0	255.255.2...	192.168.3.2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	8	192.168.7.0	255.255.2...	192.168.3.2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	9	192.168.6.0	255.255.2...	192.168.3.2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	10	192.168.5.0	255.255.2...	192.168.3.2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

ARP Table

Reset Entry

Modified	Index	IP Address	Next Hop MAC Address
<input type="checkbox"/>	0	192.168.3.2	00:00:00:00:04:04
<input type="checkbox"/>	1	192.168.15.1	00:00:00:00:0d:01
<input type="checkbox"/>	2	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	3	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	4	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	5	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	6	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	7	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	8	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	9	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	10	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	11	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	12	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	13	0.0.0.0	00:00:00:00:00:00
<input type="checkbox"/>	14	0.0.0.0	00:00:00:00:00:00

Router Control Panel

Router Quick-start

Configuration Statistics Details

Router Configuration

Interface Configuration Load From File

Port Number	MAC Address	IP Address
0	0000.0000.0004.01	192.168.6.1
1	100.00.00.00.04.02	192.168.5.2
2	200.00.00.00.04.03	192.168.4.2
3	300.00.00.00.04.04	192.168.3.2

Routing Table Reset Entry

Modified	Index	Destination IP	Subnet Mask	NextHop IP	MAC0	CPUD	MAC1	CPU1	MAC2	CPU2	MAC3	CPU3
0	192.168.15.0	255.255.2	192.168.3.1									
1	192.168.14.0	255.255.2	192.168.3.1									
2	192.168.13.0	255.255.2	192.168.3.1									
3	192.168.12.0	255.255.2	192.168.6.2									
4	192.168.11.0	255.255.2	192.168.6.2									
5	192.168.10.0	255.255.2	192.168.6.2									
6	192.168.9.0	255.255.2	192.168.6.2									
7	192.168.8.0	255.255.2	192.168.6.2									
8	192.168.7.0	255.255.2	192.168.6.2									
9	192.168.6.0	255.255.2	0.0.0.0									
10	192.168.5.0	255.255.2	0.0.0.0									

ARP Table Reset Entry

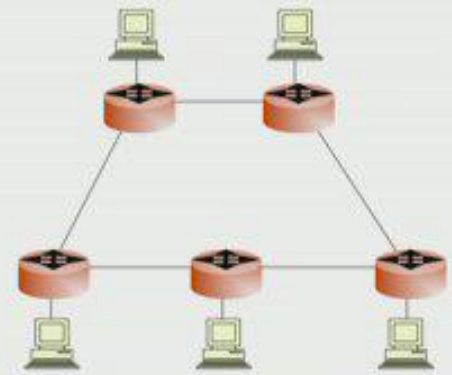
Modified	Index	IP Address	Next Hop MAC Address
0	192.168.4.1	00:15:17:60:04:00	
1	192.168.3.1	00:00:00:00:01:01	
2	192.168.6.2	00:00:00:00:07:04	
3	0.0.0.0	00:00:00:00:00:00	
4	0.0.0.0	00:00:00:00:00:00	
5	0.0.0.0	00:00:00:00:00:00	
6	0.0.0.0	00:00:00:00:00:00	
7	0.0.0.0	00:00:00:00:00:00	
8	0.0.0.0	00:00:00:00:00:00	
9	0.0.0.0	00:00:00:00:00:00	
10	0.0.0.0	00:00:00:00:00:00	
11	0.0.0.0	00:00:00:00:00:00	
12	0.0.0.0	00:00:00:00:00:00	
13	0.0.0.0	00:00:00:00:00:00	
14	0.0.0.0	00:00:00:00:00:00	
15	0.0.0.0	00:00:00:00:00:00	
16	0.0.0.0	00:00:00:00:00:00	
17	0.0.0.0	00:00:00:00:00:00	
18	0.0.0.0	00:00:00:00:00:00	

http://192.168.10.1/video/ed_hd.asf / VLC media player

Media Playback Audio Video Tools View Help

http://192.168.10.1/video/ed_hd.asf 1.00x 03:03/10:51

Streaming video



Review

NetFPGA as IPv4 router:

- Reference hardware + SCONE software
- Routing protocol discovers topology

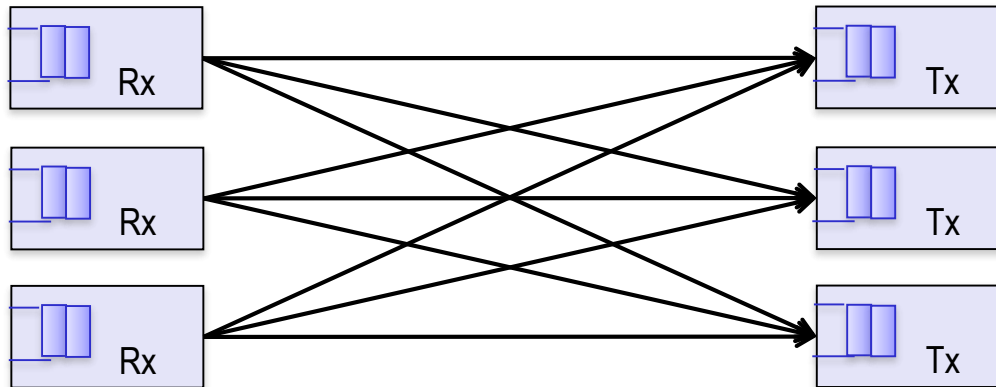
Demo:

- Ring topology
- Traffic flows over shortest path
- Broken link: automatically route around failure

Appendix II: Example II

Buffers in Routers

- **Internal Contention**
- **Pipelining**
- **Congestion**



Buffer Sizing Story



$$2T \sim C$$



$$\frac{2T \sim C}{\sqrt{n}}$$



$$O(\log W)$$

	Rule-of-thumb	Small Buffers	Tiny Buffers
# of packets	1,000,000	10,000	20 - 50
Intuition	TCP Sawtooth	Sawtooth Smoothing	Non-bursty Arrivals
Assume	Single TCP Flow, 100% Utilization	Many Flows, 100% Utilization	Paced TCP, 85-90% Utilization
Evidence	Simulation, Emulation	Simulations, Test-bed and Real Network Experiments	Simulations, Test-bed Experiments

Using NetFPGA to explore buffer size

- Need to reduce buffer size and measure occupancy
- Alas, not possible in commercial routers
- So, we will use the NetFPGA instead

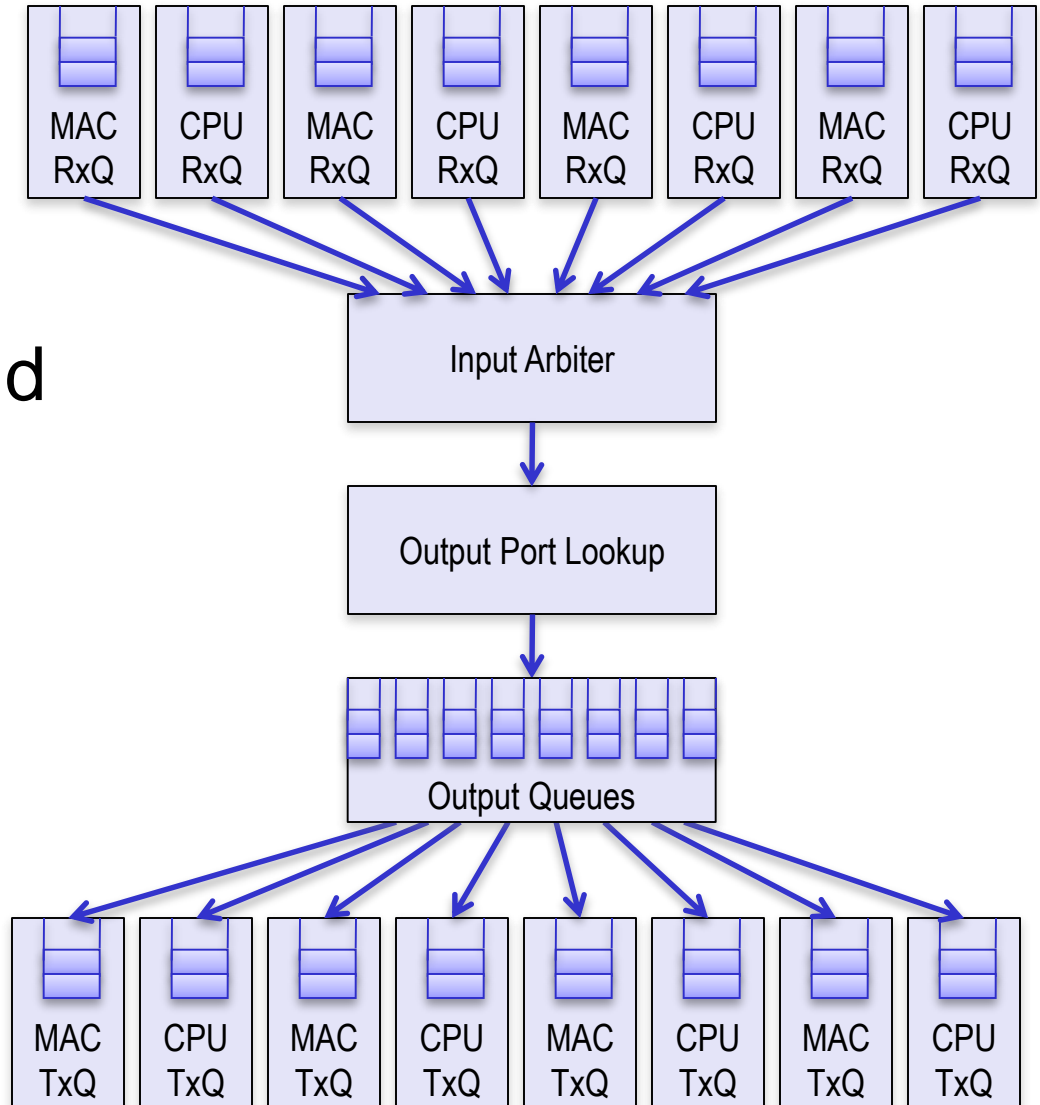
Objective:

- Use the NetFPGA to understand how large a buffer we need for a **single** TCP flow.

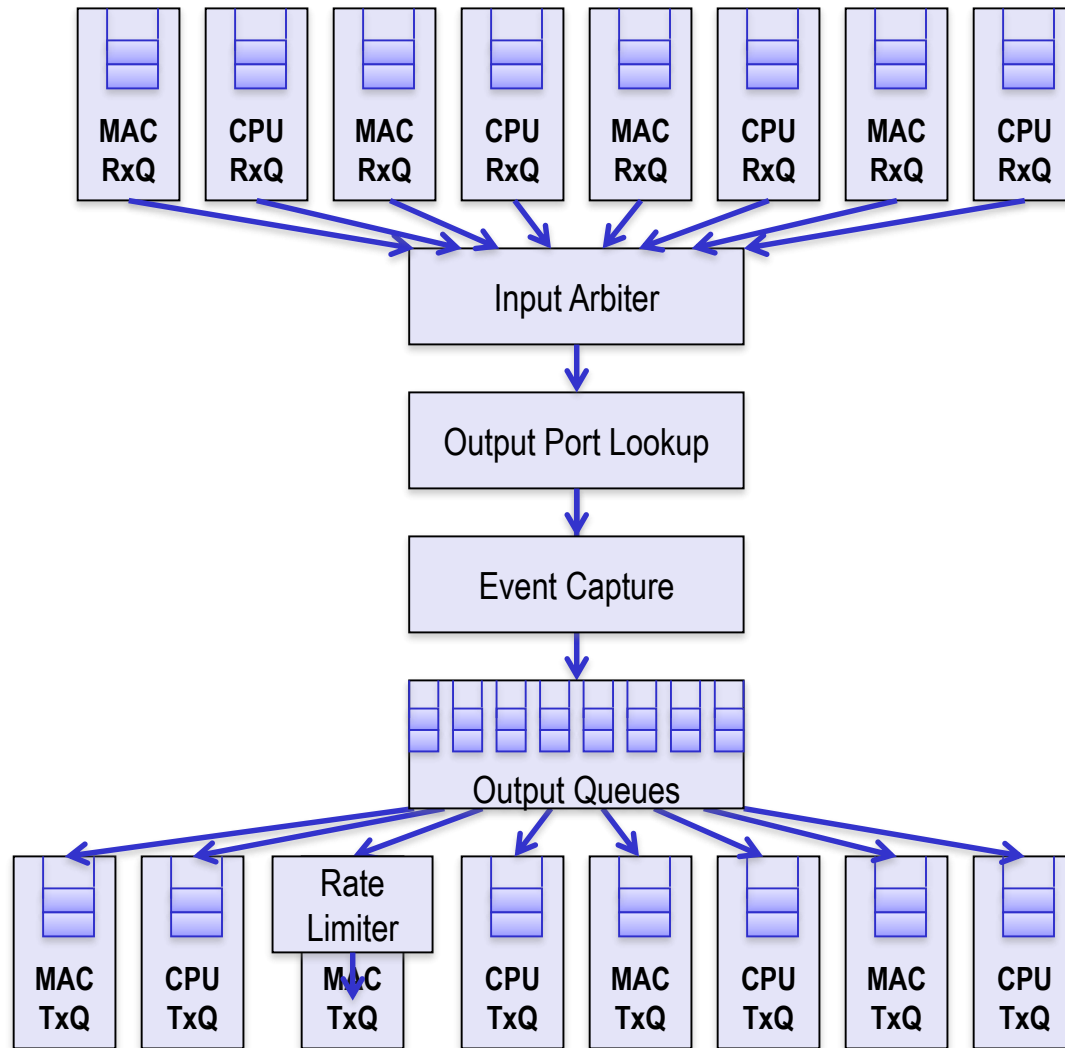


Reference Router Pipeline

- **Five stages**
 - Input interfaces
 - Input arbitration
 - Routing decision and packet modification
 - Output queuing
 - Output interfaces
- **Packet-based module interface**
- **Pluggable design**



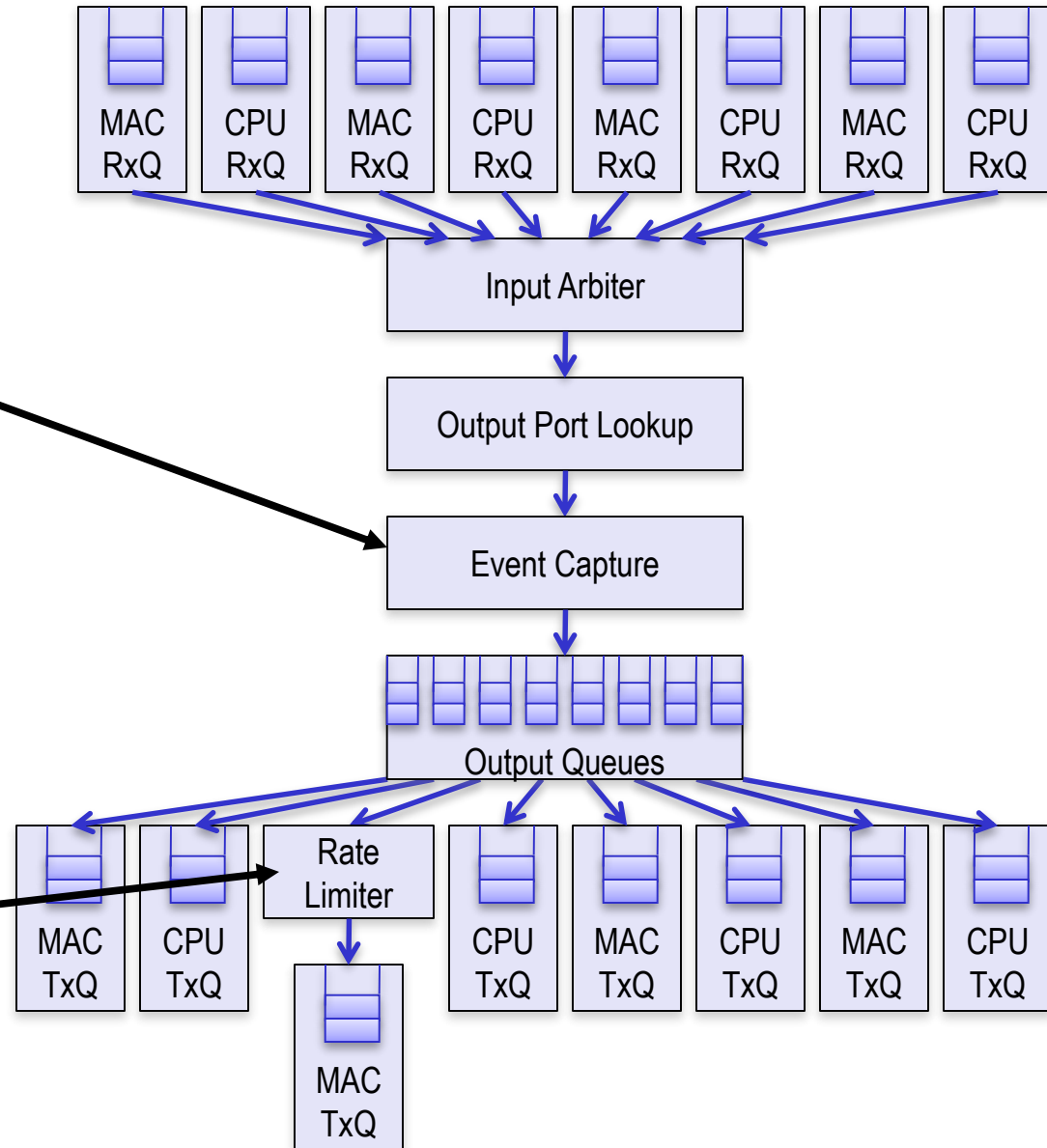
Extending the Reference Pipeline



Enhanced Router Pipeline

Two modules added

- 1. Event Capture** to capture output queue events (writes, reads, drops)
- 2. Rate Limiter** to create a bottleneck



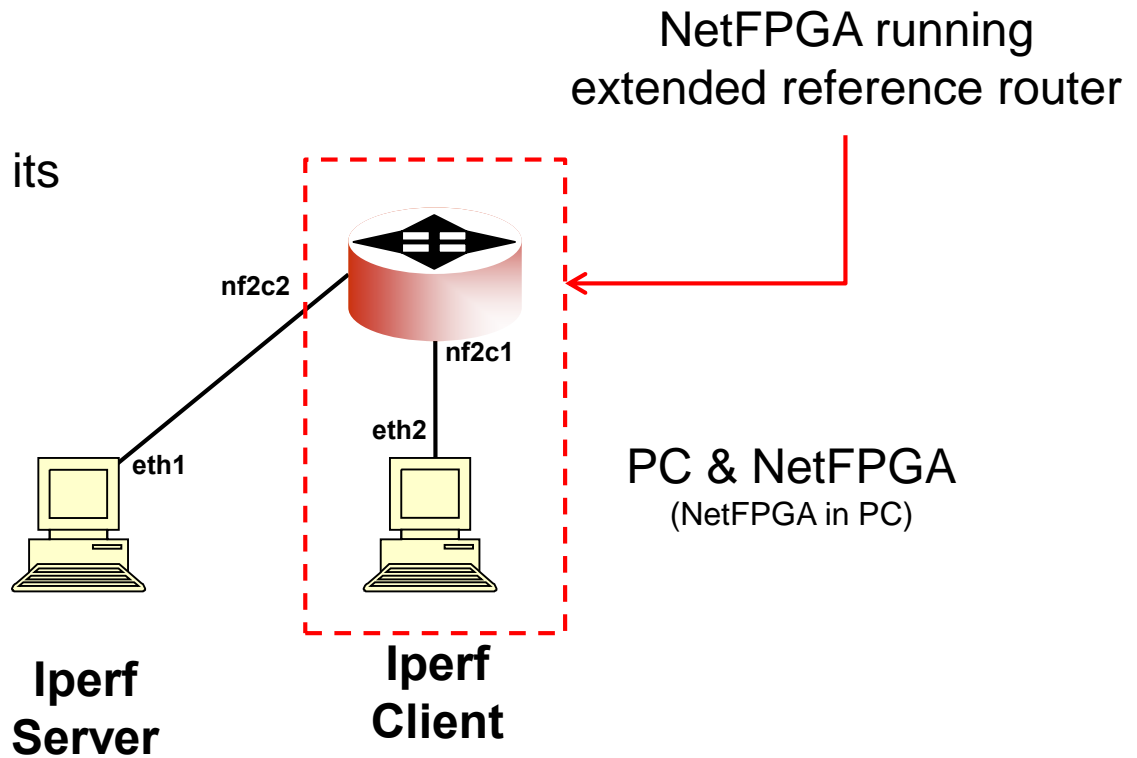
Topology for Exercise 2

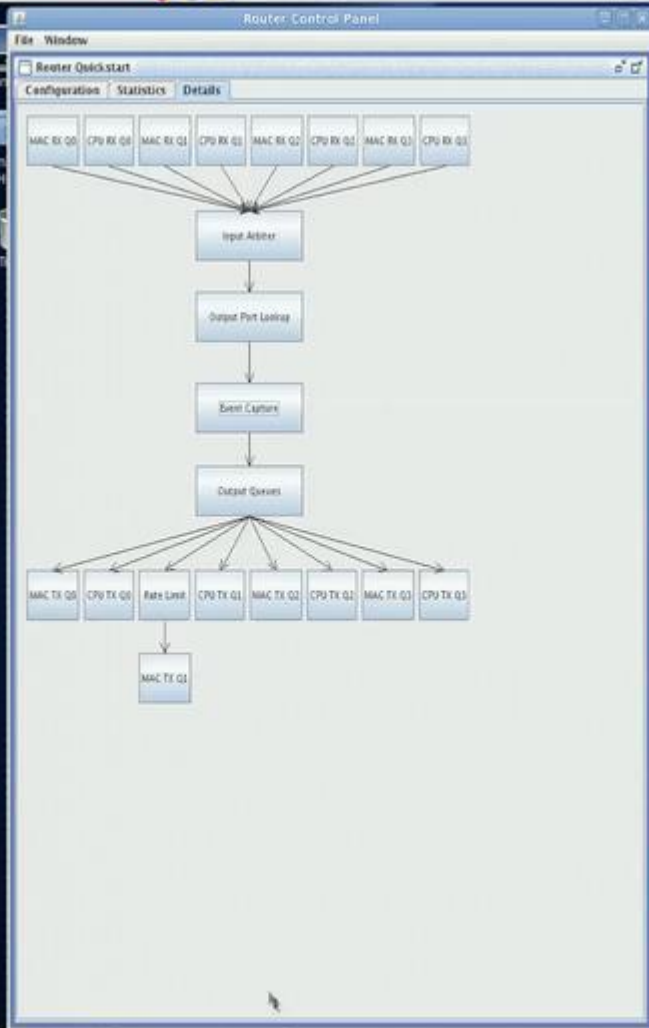
Recall:

NetFPGA host PC is life-support:
power & control

So:

The host PC may physically route its
traffic through the local NetFPGA





```

summercamp@nf-test16:~$
File Edit View Terminal Tabs Help
summercamp@nf-test16:~$
192.168.14.0 192.168.3.1 255.255.255.0 eth3 N Y
192.168.12.0 192.168.0.1 255.255.255.0 eth0 N Y
192.168.12.0 192.168.6.2 255.255.255.0 eth0 N Y
192.168.10.0 192.168.6.2 255.255.255.0 eth0 N Y
192.168.9.0 192.168.6.2 255.255.255.0 eth0 N Y
192.168.8.0 192.168.6.2 255.255.255.0 eth0 N Y
192.168.8.0 192.168.6.2 255.255.255.0 eth0 N Y
192.168.8.0 192.168.6.2 255.255.255.0 eth0 N Y
192.168.6.0 0.0.0.0 255.255.255.0 eth4 N Y
192.168.6.0 0.0.0.0 255.255.255.0 eth2 N Y
192.168.4.0 0.0.0.0 255.255.255.0 eth2 N Y
192.168.2.0 0.0.0.0 255.255.255.0 eth3 N Y
192.168.1.0 192.168.3.1 255.255.255.0 eth3 N Y

** <- Sending packet of size 114 out iface: eth0
** <- Sending packet of size 114 out iface: eth0
** <- Sending packet of size 114 out iface: eth3
** -> Received IP packet of length 66
** <- Sending packet of size 66 out iface: eth0
** <- Sending packet of size 66 out iface: eth1
** <- Sending packet of size 66 out iface: eth2
** <- Sending packet of size 66 out iface: eth3
** -> Received IP packet of length 66
    
```

Review

NetFPGA as flexible platform:

- Reference hardware + SCONE software
- new modules: event capture and rate-limiting

Example 2:

Client ↔ Router ↔ Server topology

- Observed router with new modules
- Started tcp transfer, look at queue occupancy
- Observed queue change in response to TCP ARQ