ECE 6115 / CS 8803 - ICN
Interconnection Networks for High Performance Systems
Spring 2020

INTRODUCTION

Tushar Krishna
Assistant Professor
School of Electrical and Computer Engineering
Georgia Institute of Technology

tushar@ece.gatech.edu
Background
- PhD from MIT in EECS (2013)
- Researcher at Intel (2014-15)
- Georgia Tech (2015 - present)

Office: Klaus 2318

Research Interests
- Computer Architecture
- Interconnection Networks
- Network-on-Chip
- Deep Learning Accelerators
- Continuous Learning Systems
WHAT IS AN INTERCONNECTION NETWORK?

- Networks *within* a system (collaboration)
- Not the Internet: network *between* systems
WHY INTERCONNECTION NETWORKS MATTER?

Chinese 260-core processors ShenWei SW26010 enabled supercomputer Sunway TaihuLight be the most productive in the world

IBM reveals 'brain-like' chip with 4,096 cores

Meet KiloCore, a 1,000-core processor so efficient it could run on a AA battery

IBM pushes silicon photonics with on-chip optics

Big Blue researchers have figured out how to use standard manufacturing processes to make chips with built-in optical links that can transfer 25 gigabits of data per second.
WHAT IS AN INTERCONNECTION NETWORK?

Parallel Programming/Software
- Massively Parallel Processors
- Shared Memory
- Memory Consistency
- MPI
- Infiniband
- Myrinet
- Datacenters and HPC

Many-core
- Mesh
- AMBA Bus
- High-Radix
- NIC
- System-on-Chip
- Cache Coherence Protocol
- Repeated Link
- CMOS Driver

Circuits
- Optical Waveguides
- Equalized Link

On-Chip Microarchitecture

Parallel Programming/Software

Memory Consistency

Infiniband

Myrinet

Datacenters and HPC

Many-core

Mesh

AMBA Bus

High-Radix

NIC

System-on-Chip

Cache Coherence Protocol

Repeater Link

CMOS Driver

Optical Waveguides

Equalized Link
Interconnected FPGAs form a separate plane of computation
Can be managed and used independently from the CPU
# HPC Networks

## Top 10 positions of the 50th TOP500 in November 2017[^15]

<table>
<thead>
<tr>
<th>Rank</th>
<th>Rmax (PfLOPS)</th>
<th>Name</th>
<th>Model</th>
<th>Processor</th>
<th>Interconnect</th>
<th>Vendor</th>
<th>Site country, year</th>
<th>Operating system</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>93.015 125.436</td>
<td>Sunway TaihuLight</td>
<td>Sunway MPP</td>
<td>SW26010</td>
<td>Sunway</td>
<td>NRCPC</td>
<td>National Supercomputing Center in Wuxi, China, 2016</td>
<td>Linux (Raise)</td>
</tr>
<tr>
<td>2</td>
<td>33.863 54.902</td>
<td>Tianhe-2 TH-IVB-FEP</td>
<td>Xeon E5–2692, Xeon Phi 31S1P</td>
<td>TH Express-2</td>
<td>NUDT</td>
<td>National Supercomputing Center in Guangzhou, China, 2013</td>
<td>Linux (Kylin)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>19.590 25.326</td>
<td>Piz Daint</td>
<td>Cray XC50</td>
<td>Xeon E5-2690v3, Tesla P100</td>
<td>Aries</td>
<td>Cray</td>
<td>Swiss National Supercomputing Centre Switzerland, 2016</td>
<td>Linux (CLE)</td>
</tr>
<tr>
<td>5</td>
<td>17.590 27.113</td>
<td>Titan</td>
<td>Cray XK7</td>
<td>Opteron 8274, Tesla K20X</td>
<td>Gemini</td>
<td>Cray</td>
<td>Oak Ridge National Laboratory United States, 2012</td>
<td>Linux (CLE, SLES based)</td>
</tr>
<tr>
<td>6</td>
<td>17.173 20.133</td>
<td>Sequoia Blue Gene/Q</td>
<td>A2</td>
<td>Custom</td>
<td>IBM</td>
<td>Lawrence Livermore National Laboratory United States, 2013</td>
<td>Linux (RHOL and CNK)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>14.137 43.902</td>
<td>Trinity</td>
<td>Cray XC40</td>
<td>Xeon E5–2698v3, Xeon Phi</td>
<td>Aries</td>
<td>Cray</td>
<td>Los Alamos National Laboratory United States, 2015</td>
<td>Linux (CLE)</td>
</tr>
<tr>
<td>9</td>
<td>13.555 24.914</td>
<td>Oakforest-PACS</td>
<td>Fujitsu</td>
<td>Xeon Phi 7250</td>
<td>Intel Omni-Path</td>
<td>Kashiwa, Joint Center for Advanced High Performance Computing Japan, 2016</td>
<td>Linux (CLE)</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10.510 11.280</td>
<td>K computer</td>
<td>Fujitsu</td>
<td>SPARC64 VIII fx</td>
<td>Tofu</td>
<td>Riken, Advanced Institute for Computational Science (AICS) Japan, 2011</td>
<td>Linux (CLE)</td>
<td></td>
</tr>
</tbody>
</table>

[^15]: Source of data: TOP500.org
MANY-CORE ON-CHIP NETWORKS

Sun UltraSpac T2
8 Cores

AMD MagnyCours
12 Cores

Tilera Tile64
64 Cores

IBM Power 7
8 Cores

IBM Cell
8 Cores

Intel Teraflops
80 Cores
ACCELERATOR NETWORKS

GPUs

- NVIDIA Pascal
  - 64 CUDA cores (per SM)
  - 60 SMs

Deep Learning Accelerators

- Google TPU
  - 256x256 MACs

- IBM TrueNorth
  - 4096 cores

- ShiDianNao
  - 64 PEs

- MIT Eyeriss
  - 168 PEs

- 168 PE Array

- On-chip Buffer

- NBin

- NOut

- Unified Buffer for Local Activations
  - (96Kx256x64 = 24 MIB)
  - 29% of chip

- Matrix Multiply Unit
  - (256x256x64=64K MAC)
  - 24%

- 4,096 cores
  - $10^6$ neurons
  - 256x$10^6$ synapses

- 4096 cores
  - 256 neurons
  - 65,536 synapses

- PolyMorph Engine
  - Vertex Fetch, Tessellator, Viewport Transform
  - Attribute Setup, Stream Output

- 64 KB Shared Memory / L1 Cache
  - Uniform Cache
  - Texture Cache

- Interconnect Network

- SM
  - Instruction Cache
  - Warp Scheduler
  - Dispatch Unit

- LD/ST, SFU
IOT NETWORKS
AND MANY MORE . . .

FPGA Networks

Photonic Networks

2.5D and 3D Networks
WHAT IS AN INTERCONNECTION NETWORK?

- Interconnection Networks connect processors and memory elements within and across computers
**WHAT IS AN INTERCONNECTION NETWORK?**

- **Application:** Ideally wants low-latency, high-bandwidth, dedicated channels between processors and memory

- **Technology:** Dedicated channels too expensive in terms of area and power
**Interconnection Network**: A programmable system that transports data between terminals over a set of shared physical channels.
Key Design Principles

- Transfer maximum amount of information (**high bandwidth**) within the least amount of time (**low latency**) so as to not bottleneck the system
- Efficiently utilize **shared** but **scarce resources** (buffers, links, logic) to **reduce area and power**
TYPES OF INTERCONNECTION NETWORKS

- Interconnection Networks can be grouped into four domains
  - Depending on the type and proximity of devices to be connected
- On-Chip Networks (OCNs or NoCs)
- System/Storage Area Networks (SANs)
- Local Area Networks (LANs)
- Wide Area Networks (WANs)
ON-CHIP NETWORK (OCN OR NOC)

- Networks on multicore/MPSoC chips
  - Devices include micro architectural functional units, register files, processor/IP cores, caches, directories, memory controllers

- Current/Future Systems: tens to hundreds of devices
  - Intel Single-Chip Cloud Computer – 48 Cores
  - Tilera TILE64 – 64 Cores

- Tightly coupled with on-chip links
  - Proximity: milli meters
  - Delay: pico seconds
- Multi-processor and multi-computer Networks
  - Inter-processor and processor-memory interactions

- Server and Datacenter Networks
  - Storage and I/O components

- Hundreds to thousands of devices interconnected
  - IBM Blue Gene/L Supercomputer (64K nodes, each with 2 processors)

- Tightly-coupled with proprietary interconnects
  - Proximity: tens of meters (typical) to a few hundred meters
  - Link Delay: nano seconds
  - Examples: Infiniiband, Myrinet, Quadrics, Advanced Switching Interconnect
Networks between autonomous computer systems
- Example: Machine room or throughout a building or campus
- “Clusters”

Hundreds of devices
- thousands with bridging

Loosely coupled with commodity interconnects (e.g., Ethernet)
- Proximity: few kilometers to few tens of kilometers
- Delay: micro seconds
Networks between LANs and autonomous computer systems distributed across the world

Millions of devices

Loosely coupled with electrical and optical interconnects
- Proximity: many thousands of kilometers
- Delay: milli seconds

Largest WAN is the Internet
INTERCONNECTION NETWORK DOMAINS

Source: Hennessy and Patterson, 5th Edition, Appendix F
Early designs were buses and point-to-point

DOES NOT SCALE!!!
DIFFERENCES BETWEEN OFF-CHIP (SANS) AND ON-CHIP NETWORKS

- Significant research in multi-chassis interconnection networks (off-chip) since the 90s
  - Supercomputers
  - Clusters of Workstations
  - Internet Routers

- We can leverage research insights, but …
  - constraints are different
  - new opportunities
**OFF-CHIP VS. ON-CHIP**

- **Off-Chip Networks**
  - Bandwidth limited by chip pin-bandwidth
  - Latency limited by long off-chip cables

- **On-Chip Networks**
  - Very high on-chip bandwidth
    - Abundant metal layers and wiring
    - Much lower latency due to short wires

- The key concepts remain the same across SANs and NoCs
  - The constraints are different → the design decisions are different
    - E.g., an off-chip topology may not be feasible on-chip due to on-chip layout constraints
    - Or an on-chip link circuit may not be feasible off-chip due to technology constraints

- *We will mostly focus on on-chip networks when discussing concepts, but will periodically look at implications in the off-chip space*
GENERAL-PURPOSE VS. SPECIALIZED

CMPs

Dynamic
all-to-all traffic

MPSoCs

Static fixed
traffic

DNN Accelerators

Collective Communication
AGENDA

- Course Motivation
- **Course Logistics**
- Introduction to NoCs
- Simulation Infrastructure
What’s Unique About This Course

- Not taught as a standard course in any university
  - Sits at the intersection of Computer Architecture, Parallel and Distributed Architectures, Distributed Systems, Computer Networks, and VLSI Design

- Traditional Course Structure
  - **Computer Architecture** courses focus on processor pipeline and memory hierarchy, skimming through the system interconnection
  - **Networking** courses focus on protocols, ignoring router hardware
  - **Communications** courses focus on signal processing algorithms, skimming through hardware implementations
  - **Optics/electronic link** courses focus on link circuitry, skimming through how these links are composed as networks

- Sister courses
  - **Active**: University of Toronto ECE 1749H (N Jerger)
  - **Inactive**: Past offerings in MIT (Peh), Stanford (Dally), Penn State (Das), Utah (Balasubramonian), Cornell (Batten)
WHAT'S UNIQUE ABOUT THIS COURSE

- Handful of top researchers in NoCs across academia and industry
  - opportunity to gain expertise in a niche area
    - aka internships/jobs of students who take this class
    - manycore architectures (Intel, AMD, IBM)
    - GPUs and Accelerators (NVIDIA, ARM, Intel, Samsung)
    - supercomputers (IBM, NVIDIA, Cray)
    - datacenters (Google, Amazon, Facebook, Microsoft)
    - internet routers (Cisco, Juniper)

- Projects will be open-research questions
  - Very high chance of publication!
  - 2016 Version: HPCA, ISPASS, ICCAD, NOCS
  - 2017 Version: ASPLOS, ISPASS, MICRO
  - 2018 Version: NOCS, ICRC, ISPASS*
  - 2019 Version: NOCS, ISPASS*
    - (*under submission)
COURSE STRUCTURE

- Phase I [Jan-Feb]
  - ~7-8 Instructor Lectures
  - 4 Programming Lab Assignments
  - One Midterm Quiz

- Phase II [Mar-Apr]
  - Paper Readings, Critiques, and Discussions
  - Each student presents one paper in class and leads its discussion

- Phase III [End of Feb – Apr]
  - Research Project
## Grading

<table>
<thead>
<tr>
<th>Item</th>
<th>Percentage</th>
<th>Phase I</th>
<th>Phase II</th>
<th>Phase III</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lab 1</td>
<td>3%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lab 2</td>
<td>10%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lab 3</td>
<td>10%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lab 4</td>
<td>10%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Midterm Quiz</td>
<td>10%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paper Critiques</td>
<td>10% [Best of 10]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paper Presentation</td>
<td>10%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project - Proposal</td>
<td>5%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project - Milestones</td>
<td>10%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project - Presentation</td>
<td>10%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project - Final Report</td>
<td>12%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>100%</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
PHASE I

- Lectures
  - ~7-8 weeks of instructor lectures

- Required Textbook
    - Available for free download (within Georgia Tech)
    - Also added to Canvas
  - Supplemental reading material/papers will be posted

- Optional Textbooks:
Lab Assignments

- C++ based programming assignments
  - Build various components of a network, run traffic, analyze performance and costs

- Familiarize yourself with a network simulator called Garnet2.0
  - Part of gem5 (www.gem5.org), one of the leading open-source simulators for computer architecture research in industry and academia [Over 2000 citations in 6 years]

- Setup inside GT:
  - [http://tusharkrishna.ece.gatech.edu/teaching/garnet_gt/](http://tusharkrishna.ece.gatech.edu/teaching/garnet_gt/)
PHASE I

- **Lab 1 is due Friday at 1:00 pm**
  - **Hard Deadline**
  - **Already up on Canvas**
    - Gives you time to drop course if you don’t have right background
  - **Goal:** run synthetic traffic simulations for 4 traffic patterns and report the latency vs. injection rate results
    - *Setup Garnet right away and get started!*
PHASE I

- **Midterm Quiz**
  - Short in-class Quiz on the topics discussed in class
  - Aim is to make sure you go back and read the slides after each lecture
Writing, Presenting, and Discussing research ideas will be key thrusts.

In every class, we will discuss 2-3 papers:

- State-of-the-art research across a breadth of domains
  - Datacenters, HPC, On-Chip, GPU, TPU, FPGAs, Circuits, Novel Technologies

Everyone (including presenter) has to submit a 1 page write-up before the beginning of class on one of the papers:

- Short Summary + 2 strengths + 2 weaknesses + 1 suggested improvement

One student will present on one of the papers/topics for 15-20 minutes and lead the class discussion:

- Make your own slides
  - The presenter should create a Piazza post for the paper when it is assigned.
- Might have 2-3 presentations per class on different papers taking opposing views, leading to a healthy debate.
PHASE III

- Research Project
  - Propose a solution for a research problem in the Network-on-Chip/System-Area Network space; implement and evaluate it
    - Implement an idea from a paper and propose an extension, or implement a completely novel idea
    - Start thinking of project ideas as I present topics in class
      - I will also periodically provide a list of potential ideas
    - Groups of 2
### Project Ideas

**Deep Learning**
- Performance evaluation of Google’s TPU systolic array (released in 2017)
- Performance characterization of NVIDIA’s NVDLA Network (released in 2017)
- NoC Topologies for efficient mapping strategies
- NoC for scale-out DNN accelerators
- NoCs for FPGA-based Deep Learning Accelerators

**Microarchitecture**
- SMART NoC emulating High-Radix Topology
- NoCs for heterogeneous CPU-GPU systems
- NoCs inside GPUs
- Approximation-aware NoC
- 2.5D Networks on a package using different packaging technologies

**Open-source Hardware (in RTL/Chisel)**
- Network-on-Chip in Chisel
- NoC in Verilog [build upon prior work]

**Cloud and Edge Networks**
- Networks on a Rack
- Wireless Networks between Raspberry Pis
PHASE III

- Research Project
  - Infrastructure
    - The project can be implemented in Garnet, or any other tool (C++/RTL)
      - Garnet/gem5 is useful since the plumbing related to other parts of the system (and even real apps) are provided.
    - We will introduce you to Chisel and Bluespec System Verilog (C-like abstractions for generating Verilog)
  - Projects related to your own Special Problem / MS / PhD research work will be encouraged as long as they have a networks component
  - Projects can be done individually (preferred) or in groups of two (if scope is larger, clearly defining each member’s role)
PHASE III

- Research Project Milestones
  - Proposal Presentation
  - Progress Milestone #1
  - Progress Milestone #2
  - Final Presentation
    - No Final Exam!
  - Final Report
HOW PROJECTS WILL BE EVALUATED

- Looking for thorough understanding, implementation, evaluation, and presenting of idea
  - the idea might not lead to any improvement over the state-of-the-art
    - that is OK
    - rather that is research!

- If the idea leads to novel insights and there is a chance for publication, I am excited to work with you on polishing the work over summer and submit it to a conference or journal
  - I will fund your travel for presenting at the conference if it gets accepted 😊
  - If you want to work on a novel publishable idea or an extension to your own MS/PhD research, contact me – part of it can be used for the course
# Schedule (Tentative)

<table>
<thead>
<tr>
<th>Week</th>
<th>Dates</th>
<th>Monday</th>
<th>Wednesday</th>
<th>Due [Friday]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(Jan 6 - )</td>
<td></td>
<td></td>
<td>Lab 1</td>
</tr>
<tr>
<td>2</td>
<td>(Jan 11 - )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(Jan 20 - )</td>
<td></td>
<td>MLK Day</td>
<td>Lab 2</td>
</tr>
<tr>
<td>4</td>
<td>(Jan 27 - )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(Feb 3 - )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>(Feb 10 - )</td>
<td></td>
<td></td>
<td>Lab 3</td>
</tr>
<tr>
<td>7</td>
<td>(Feb 17 - )</td>
<td></td>
<td></td>
<td>Project Proposals</td>
</tr>
<tr>
<td>8</td>
<td>(Feb 24 - )</td>
<td></td>
<td>Midterm</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>(Mar 2 - )</td>
<td></td>
<td></td>
<td>Lab 4</td>
</tr>
<tr>
<td>10</td>
<td>(Mar 9 - )</td>
<td></td>
<td></td>
<td>Proposal Ppt</td>
</tr>
<tr>
<td>11</td>
<td>(Mar 16 - )</td>
<td>Spring Break</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>(Mar 23 - )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>(Mar 30 - )</td>
<td></td>
<td></td>
<td>Milestone 1</td>
</tr>
<tr>
<td>14</td>
<td>(Apr 6 - )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>(Apr 13 - )</td>
<td></td>
<td></td>
<td>Milestone 2</td>
</tr>
<tr>
<td>16</td>
<td>(Apr 20 - )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>(Apr 27 - )</td>
<td>Final Presentations</td>
<td></td>
<td>Final Report</td>
</tr>
</tbody>
</table>

GT Holiday
Instructor Travel
Phase I
Phase II
Phase III
Course Information

- **Course Website**
  - [http://tusharkrishna.ece.gatech.edu/teaching/icn_s20/](http://tusharkrishna.ece.gatech.edu/teaching/icn_s20/)
  - All additional readings will be posted here!

- **Canvas**
  - Lecture Slides will be posted here
  - Lab Assignments will be posted and submitted here
    - Lab 1 is also posted on website for wait-listed students
  - Project Reports will be submitted here

- **Piazza**
  - Access via Canvas
  - **Use for questions related to the lab assignments**
    - Try to answer each other’s questions
    - **There are no TAs in this course**
    - Do not upload code!
    - If no one is able to answer within a day I will respond
  - **Discussions on paper readings also encouraged**
WHAT IS EXPECTED FROM YOU?

- **Required Background**
  - This is an **advanced** graduate-level class
  - Graduate-level Computer Architecture (ECE 6100 / CS 6290) background expected
  - C++ Programming Knowledge
    - Labs + Project

- **Willingness to do explore open research problems!**
  - Heavy paper reading [~15 papers]
  - Lot of Writing
    - Critiques, Reviews, and Report
  - Identify, Implement, Evaluate and Present a new idea
  - 3 presentations (Paper + Proposal + Final Report)
HOW TO CONTACT ME

- Piazza

- Email for questions not suitable for Piazza
  - Why did I lose points?
  - Is this an acceptable project proposal

- Office Hours
  - Friday 1:00 – 2 PM in Klaus 2318
AGENDA

- Course Motivation
- Course Logistics
- Introduction to NoCs
- Simulation Infrastructure
INTRODUCTION TO NOCS

On-Chip Network

Core + L1$

Core + L1$

Core + L1$

Core + L1$

Core + L1$

Core + L1$

L2$

L2$

L2$

L2$

L2$

L2$
ROLE OF THE NETWORK-ON-CHIP

- “Shared Memory” Systems
  - Transport Cache Lines and Cache Coherence Messages between caches and memory controller(s) in shared memory CMPs

- “Message Passing” Systems
  - Transfer data between IP blocks in MPSoCs (Multi-Processor System on Chip)
On-Chip Network

If network delay = 10 cycles, controller delay = 5 cycles, how many cycles before data arrives?
Core will not be shown explicitly in the most of the slides. Only the routers will be.
NETWORK ARCHITECTURE

- Topology
- Routing
- Flow Control
- Router Microarchitecture
TOPOLOGY: HOW TO CONNECT THE NODES WITH LINKS

~Road Network
ROUTING: WHICH PATH SHOULD A MESSAGE TAKE

~Series of road segments from source to destination
FLOW CONTROL: WHEN DOES THE MESSAGE STOP/PROCEED

~Traffic Signals / Stop signs at end of each road

[Map showing traffic signals and roads in a city]
ROUTER MICROARCHITECTURE: HOW TO BUILD THE ROUTERS

~Design of traffic intersection (number of lanes, algorithm for turning red/green)
Oracle SPARC T5 (2013)
16 multi-threaded cores and 8 L2 banks
connected by a Crossbar NoC

IBM Cell (2005)
1 general purpose and 8 special purpose engines
connected by a Ring NoC

Intel SCC (2009)
24 tiles with 2 cores each
connected by a Mesh NoC
Network resources are distributed with almost no centralized control

Traffic is (often) unpredictable

**Why?**
- Mapping of tasks to cores
- Memory layout of data
- Cache sizes, policies
- Data sharing
- ...

Surprisingly easy for the network to become the bottleneck
NOC METRICS

- Performance
  - Latency
  - Bandwidth

- Power
  - Energy Consumption in Links
  - Energy Consumption in Routers

- Area
HOW TO EVALUATE A NETWORK?

Latency

Offered Traffic (bits/sec)
AGENDA

- Course Motivation
- Course Logistics
- Introduction to NoCs
- Simulation Infrastructure
**THE GEM5 FULL-SYSTEM SIMULATOR**

http://www.gem5.org  Join the mailing list!

http://tusharkrishna.ece.gatech.edu/teaching/garnet_gt/
has instructions on setup for this class

<table>
<thead>
<tr>
<th>Workload</th>
<th>Simple (Fast)</th>
<th>Detailed (Slow)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS</td>
<td>System Emulation</td>
<td>Full-System</td>
</tr>
<tr>
<td>ISA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPU</td>
<td>AtomicSimple</td>
<td>InOrder</td>
</tr>
<tr>
<td></td>
<td>TimingSimple</td>
<td>OutOfOrder</td>
</tr>
<tr>
<td>CPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L1</td>
<td>Classic</td>
<td>Ruby</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* Caches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* Coherence protocols</td>
</tr>
<tr>
<td>L2+Dir</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L1</td>
<td></td>
<td>Garnet2.0</td>
</tr>
<tr>
<td>L2+Dir</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L2+Dir</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L2+Dir</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network-on-Chip</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
User can inject any traffic pattern

Sources: CPUs
Destinations: Directories
Traffic Pattern determines which CPU sends to which Directory.
Injection Rate is user specified

Directories consume any packet that is received.

Source (binary coordinates):
\[(y_{k-1}, y_{k-2}, \ldots, y_1, y_0, x_{k-1}, x_{k-2}, \ldots, x_1, x_0)\]

<table>
<thead>
<tr>
<th>Traffic Pattern</th>
<th>Destination (binary coordinates)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit-Complement</td>
<td>(\bar{y}<em>{k-1}, \bar{y}</em>{k-2}, \ldots, \bar{y}<em>1, \bar{y}<em>0,) (\bar{x}</em>{k-1}, \bar{x}</em>{k-2}, \ldots, \bar{x}_1, \bar{x}_0)</td>
</tr>
<tr>
<td>Bit-Reverse</td>
<td>(x_0, x_1, \ldots, x_{k-2}, x_{k-1}, y_0, y_1, \ldots, y_{k-2}, y_{k-1})</td>
</tr>
<tr>
<td>Shuffle</td>
<td>(y_{k-2}, y_{k-3}, \ldots, y_0, x_{k-1}, x_{k-2}, x_{k-3}, \ldots, x_0, y_{k-1})</td>
</tr>
<tr>
<td>Tornado</td>
<td>(y_{k-1}, y_{k-2}, \ldots, y_1, y_0, x_{k-1}) (x_{k-1 + \lfloor \frac{k}{2} \rfloor - 1}, \ldots, x_{\lfloor \frac{k}{2} \rfloor - 1})</td>
</tr>
<tr>
<td>Transpose</td>
<td>(x_{k-1}, x_{k-2}, \ldots, x_1, x_0, y_{k-1}, y_{k-2}, \ldots, y_1, y_0)</td>
</tr>
<tr>
<td>Uniform Random</td>
<td>\textit{random()}</td>
</tr>
</tbody>
</table>
If you have random traffic, how many hops does each message take on average on this 4x4 mesh topology?

*Can you generalize this to a $k \times k$ mesh?*