ECE 8823 A / CS 8803 - ICN
Interconnection Networks for High Performance Systems
Spring 2019

TOPOLOGY

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

tushar@ece.gatech.edu
RECAP

On-Chip Network / Network-on-Chip

System/Storage Area Network
WHY NOCS ARE IMPORTANT
NETWORK ARCHITECTURE

- **Topology**
  - How to connect the nodes
  - Road Network

- **Routing**
  - Which path should a message take
  - Series of road segments from source to destination

- **Flow Control**
  - When does the message have to stop/proceed
  - Traffic signals at end of each road segment

- **Router Microarchitecture**
  - How to build the routers
  - Design of traffic intersection (number of lanes, algorithm for turning red/green)
**NETWORK PERFORMANCE**

*Saturation Throughput: the injection rate at which latency ~3x(zero-load latency)*

- **Zero load latency** (topology+routing+flow control)
- **Min latency given by routing algorithm**
- **Min latency given by topology**
- **Throughput given by flow control**
- **Throughput given by routing**
- **Throughput given by topology**

```
Offered Traffic (bits/sec)
```
TOPOLOGY:
HOW TO CONNECT THE NODES WITH LINKS

~Road Network
**TOPOLOGY OVERVIEW**

- Often the first step in network design

- Significant impact on network cost-performance
  - Determines implementation complexity, i.e., **cost**
    - number of routers and links
    - router degree (i.e., ports)
    - ease of layout
  - Determines application **performance**
    - number of hops $\to$ latency and energy consumption
    - maximum throughput
**HOW TO SELECT A TOPOLOGY?**

**Application’s Task Communication Graph**
- Vertices: tasks
- Edges: communication

**Network Topology Graph**
- Vertices: cores
- Edges: links

**Best topology?**

**Problems?**
- Cannot change algorithm
- Cannot change mapping
- Cannot adapt to data-dependent load imbalance in application
- Layout/packaging issue with long wires and high-node degree

*Topology is fixed at design-time.*

**Benefits to being regular and flexible**
LET US DEFINE SOME DESIGN-TIME METRICS

- **Degree** – number of ports at a node
  - Proxy for area/energy cost

- **Bisection Bandwidth** - bandwidth crossing a minimal cut that divides the network in half
  - (Min # channels crossing two halves) * (BW of each channel)
  - Proxy for peak bandwidth
    - Can be misleading as it does not account for routing and flow control efficiency
    - At this stage, we assume **ideal routing** (perfect load balancing) and **ideal flow control** (no idle cycles on any channel)

- **Diameter** – maximum routing distance (number of links in *shortest* route)
  - Proxy for latency
SOME *RUN-TIME* METRICS

- **Hop count (or routing distance)**
  - Number of hops between a communicating pair
  - Depends on application and mapping
  - *Average hop count or Average distance*: average hops across all valid routes

- **Channel load**
  - Number of flows passing through a particular link
  - Depends on application and mapping
  - *Maximum channel load determines throughput*

- **Path diversity**
  - Number of shortest paths between a communicating pair
    - Can be exploited by routing algorithm
    - Provides *fault tolerance*
Can you suggest a regular topology (each router with same degree) with smallest possible diameter?

**Trick question :p**
One node. Degree = 0, Diameter = 0.

**Application’s Task Communication Graph**

Vertices - tasks
Edges - communication
Can you suggest a regular topology (each router with same degree) with **diameter = 1**?

**Application’s Task Communication Graph**

- Vertices - tasks
- Edges - communication

**Challenge?**
Not scalable!!
Cannot layout more than 4-6 cores in this manner for area and power reasons

**Fully Connected**
- Degree = 5
- Bisection BW = 9
### Pros
- Cost-effective for small number of nodes
- Easy to implement snoopy coherence
- Most multicore systems with 4-6 cores use Buses

### Cons
- Bandwidth! → Not scalable
POPULAR BUS PROTOCOLS

- ARM AMBA Bus
  - AHB
  - AXI
  - ACE
  - CHI

- IBM Core Connect

- ST Microelectronics STBus

- How to increase bus bandwidth?
  - Hierarchical Buses
  - Split-buses
ROUTE PACKETS, NOT WIRES!

Abstract

Using on-chip interconnection networks in place of all-the-globally-wired structures is the top-level wire on a chip and facilitates modular design. With this approach, system modules (processors, memories, peripherals, etc.) communicate by sending packets to one another over the network. The structured network wiring gives well-controlled electrical parameters that eliminate timing iterations and enable the use of high-performance circuits to reduce latency and increase bandwidth. The area overhead required to implement an on-chip network is modest, we estimate 6.6%. This paper introduces the concept of on-chip networks, sketches a simple network, and discusses some challenges in the architecture and design of these networks.

1 Introduction

We propose replacing design-specific global on-chip wiring with a general-purpose on-chip interconnection network. As shown in Figure 1, a chip employing an on-chip network is composed of a number of network clients: processor, DRAM, memory, peripherals, etc. Instead of connecting these top-level modules by routing dedicated wires, they are connected to a network that routes packets between them. Each client is placed in a rectangular tile on the chip and communicates with all other clients, not just its neighbors, via the network. The network logic occupies a small amount of area (we estimate 6.6%) in each tile and makes use of a portion of the upper two wiring layers.

Using a network to replace global wiring has advantages of structuring, performance, and modularity. The on-chip network structures the global wires so that their electrical properties are optimized and well-controlled. These controlled electrical parameters, in particular low and predictable crosstalk, enable the use of aggressive signaling circuits that can reduce power dissipation by a factor of two and increase propagation velocity by three times [3]. Sharing the wiring resources between many communication flows makes more efficient use of the wires; when one client is idle, other clients continue to make use of the network resources.

An on-chip interconnection network facilitates modularity by defining a standard interface in much the same manner as a backplane bus. For the past three decades backplanes have been common. The birth of “on-chip” switched networks

Dally and Towles, DAC 2001

The birth of “on-chip” switched networks

Figure 1: Partitioning the die into module tiles and network logic
**TOPOLOGY CLASSIFICATION**

- **Direct**
  - Each router (switch) is associated with a terminal node
  - All routers are sources and destinations of traffic
  - Example: Ring, Mesh, Torus
    - Most on-chip networks use direct topologies

- **Indirect**
  - Routers (switches) are distinct from terminal nodes
  - Terminal nodes can source / sink traffic
  - Intermediate nodes switch traffic
  - Examples: Crossbar, Butterfly, Clos, Omega, Benes, …
    - Next lecture
**RING AND TORUS**

- Formally: $k$-ary $n$-cube
  - $k^n$ network nodes
  - $n$-dimensional grid with $k$ nodes in each dimension

---

8-ary 1-cube

4-ary 2-cube
**RING**

- **Pros**
  - Cheap: $O(N)$ cost
  - Used in most multicores today

- **Cons**
  - High latency
  - Difficult to scale – bisection bandwidth remains constant
  - No path diversity
    - 1 shortest path from A to B

| Diameter? | N/2 |
| Avg Distance? | N/4 |
| Bisection BW? | 2 |
| Degree? | 2 |
Pros

- $O(N)$ cost
- Exploit locality for near-neighbor traffic
- High path diversity
  - 6 shortest paths from A to B
- Edge symmetric
  - good for load balancing
  - Same router degree

Cons

- Unequal link lengths
- Harder to layout

Diameter? $\sqrt{N}$
Bisection BW? $2\sqrt{N}$
Degree? 4
### MESH

- **Pros**
  - $O(N)$ cost
  - Easy to layout on-chip: regular and equal-length links
  - Path diversity
    - 3 shortest paths from A to B

- **Cons**
  - Not symmetric on edges
  - Performance sensitive to placement on edge vs. middle
  - Different degrees for edge vs. middle routers
  - Blocking, i.e., certain paths can block others (unlike crossbar)

<table>
<thead>
<tr>
<th>Diameter?</th>
<th>$2(\sqrt{N}-1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bisection BW?</td>
<td>$\sqrt{N}$</td>
</tr>
<tr>
<td>Degree?</td>
<td>4</td>
</tr>
</tbody>
</table>
FOLDED TORUS

Easier to layout

Is there any con compared to the mesh?

*All channels have double the length*
MULTI-DIMENSIONAL TOPOLOGIES

- Used in Supercomputers, Datacenters, and other off-chip System Area Networks

- Example:

2,3,4-ary 3 Mesh
RUN-TIME METRICS

- Hop Count
  - Latency

- Maximum Channel Load
  - Throughput
We will consider Uniform Random Traffic

Max = 4
Avg = 2.22

Max = 2
Avg = 1.33

Max = 4
Avg = 1.77

\[ H_{\text{avg}} = \begin{cases} \frac{nk}{4} & \text{k even} \\ n\left(\frac{k}{4} - \frac{1}{4k}\right) & \text{k odd} \end{cases} \]
**NETWORK LATENCY**

- $T = H \cdot t_r + T_w + T_s + T_c$
  - $H =$ number of hops
  - $t_r =$ router delay
  - $T_w =$ wire delay
  - $T_s =$ serialization delay
  - $T_c =$ contention delay

- $T = H \cdot t_r + \frac{D}{v} + \frac{L}{b} + T_c$
  - $D =$ wire distance
  - $v =$ propagation velocity
  - $L =$ packet length
  - $b =$ channel bandwidth
HOW TO REDUCE HOP COUNT?

- Low-diameter topology
  - Challenge?
    - high-radix of each switch

- Some dedicated long-range links
  - High-radix for few switches
  - How to decide where to add long links?
ST SPIDERGON

- Proprietary NoC from ST Microelectronics
- Pseudo-regular topology
  - All routers have 2 or 3 ports
  - Depending on application BW needs, links can be added removed
    - e.g., (a) vs. (b), and (c) vs. (d)
  - Easy to layout
SMALL WORLD NETWORKS

- Milgram’s Experiment and “Six degrees of separation”
  - Common across neurons, WWW, electrical power grid, ...

- Add few long-distance links to a mesh randomly reduces average distance $\sim \log N$
  - “It’s a Small World After All’: NoC Performance Optimization Via Long-Range Link Insertion”, VLSI 2006
RUN-TIME METRICS

- Hop Count
  - Latency
- Maximum Channel Load
  - Throughput

- We will consider Uniform Random Traffic
MAXIMUM CHANNEL LOAD

- Identify channel with maximum traffic
  - Count total flows through it

- Maximum Throughput = $1 / (\text{max channel load})$
Identify bottleneck channel

- For uniform random traffic, is the bisection channel

Suppose each node generates p messages per cycle

- 4p messages per cycle in left ring
- 2p message per cycle will cross to other ring
- Link can handle one message per cycle
- So maximum injection rate of p = \( \frac{1}{2} \)
What if Hot Spot Traffic?
- Suppose every node sends to node G

Which is the bottleneck channel?
- Used by A, B, C, D, E, and F to send to G
- Max Throughput = 1 / 6
With uniform random traffic
- 3 sends 1/8 of its traffic to 4,5,6
- 3 sends 1/16 of its traffic to 7 (2 possible shortest paths)
- 2 sends 1/8 of its traffic to 4,5
- Etc

Max Channel load = 1
### TRAFFIC PATTERNS

<table>
<thead>
<tr>
<th>Traffic Pattern</th>
<th>Destination (binary coordinates)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit-Complement</td>
<td>$\left( \overline{y}<em>{k-1}, \overline{y}</em>{k-2}, \ldots, \overline{y}<em>1, \overline{y}<em>0, \right.$ $\overline{x}</em>{k-1}, \overline{x}</em>{k-2}, \ldots, \overline{x}_1, \overline{x}_0 \left. \right)$</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>$(y_{k-1}, y_{k-2}, \ldots, y_1, y_0,$ $x_{k-1+\left[ \frac{k}{2} \right]-1}, \ldots, x_{\left[ \frac{k}{2} \right]-1})$</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>$\text{random()}$</td>
</tr>
</tbody>
</table>

- Historically derived from particular applications of interest.
- Important to stress test the network with different patterns.
  - Uniform random can make bad topologies look good.
- For a particular topology and traffic pattern, one can derive:
  - Avg Hop Count (→ Low-Load Latency)
  - Max Channel Load (→ Peak Throughput)
IS IT POSSIBLE TO ACHIEVE DERIVED LOW-LOAD LATENCY & PEAK THROUGHPUT?

Latency

Zero load latency (topology+routing+flow control)

Min latency given by routing algorithm

Min latency given by topology

Throughput given by flow control

Throughput given by routing

Throughput given by topology

Offered Traffic (bits/sec)
**Zero-load latency?**

(“Ideal Latency”)

\[
T = (H+1). (t_{\text{router}} + t_{\text{stall\_avg}}) + (H+2). (t_{\text{wire}}) + T_{\text{ser}}
\]

- \(H\) = number of hops inside network
- \(t_{\text{router}}\) = per-hop router pipeline delay
- \(t_{\text{wire}}\) = per-hop link delay
- \(t_{\text{stall}}\) = per-hop stall delay (due to contention)
- \(T_{\text{ser}}\) = serialization delay

\[
H_{\text{avg}} = \begin{cases} 
\frac{nk}{3} & \text{k even} \\
\frac{n(k^2 - \frac{1}{3k})}{3} & \text{k odd}
\end{cases}
\]

**Ideal case:** \(t_{\text{router}} = 1, t_{\text{wire}} = 1\)

*Let’s assume 1-flit packets (\(T_{\text{ser}} = 0\))*

Zero-load => \(t_{\text{stall\_avg}} \sim 0\)

Suppose \(k = 8\), \(H_{\text{avg}} = 5.333\) => \(T_{\text{zero\_load}} = 13.666\)
Saturation Throughput?  
(“Ideal Throughput” or Peak Injection Rate)

\[
\frac{1}{\text{max channel load}}
\]

Let's calculate load on one of the bisection links

- \(k^2/2\) nodes on the left.
- Half their messages (\(k^2/4\)) cross the bisection links.
- Total \(k\) bisection links from left to right.
- Load on each bisection link = \(k^2/4k = k/4\)

\[\text{Peak Throughput} = \frac{4}{k}\]

For \(k = 4\), peak throughput = 1 flit/node/cycle
For \(k = 8\) (64-core mesh), peak throughput = \(1/2\) flits/node/cycle
ANOTHER REPRESENTATION OF PERFORMANCE: INJECTION RATE AS A % OF “CAPACITY”

For 4x4 Mesh, 100 => 1 flit/node/cycle
For 8x8 Mesh, 100% => 0.5 flits/node/cycle

This representation is better to understand if we are able to achieve the throughput the network was actually designed for.
# Topology Classification

## Direct
- Each router is associated with a terminal node
- All routers are sources and destinations of traffic
- Example: Ring, Mesh, Torus
  - Most on-chip networks use direct topologies

## Indirect
- Routers are distinct from terminal nodes
- Terminal nodes can source / sink traffic
- Intermediate nodes switch traffic
- Examples: Crossbar, Butterfly, Clos, Omega, Benes, …
### Pros
- Every node connected to all others (**non-blocking**)
- Low latency and high bandwidth
- Used by GPUs

### Cons
- Area and Power goes up quadratically ($O(N^2)$ cost)
- Expensive to layout
- Difficult to arbitrate

---

Diameter = ? 1
Degree = ? 1
Bisection BW = ? $N$
As a convention, source and destination nodes drawn logically separate on the left and right, though physically the two 0s, two 1s, etc are often the same physical node.

Radix of each switch = \( k \) (i.e., \( k \) inputs and \( k \) outputs)

Number of stages = \( n \)

Total Source/Destination Terminal Nodes = \( k^n \)

In each stage, \( k^{n-1} \) switches
Each switch is a \( k \times k \) crossbar
**BUTTERFLY (K-ARY N-FLY): METRICS**

- **Degree?**  
  \[ k \]

- **Diameter?**  
  \[ n + 1 \]

- **Bisection Bandwidth?**  
  \[ \frac{N}{4} \]
  where \( N = k^n \)

- **Hop Count?**  
  \[ n + 1 \]

- **Channel Load?** (for uniform traffic)  
  \[ 1 \]

- **Path Diversity?**  
  None. Only one route between any pair
TACKLING PATH DIVERSITY IN A BUTTERFLY

Additional Stage
Pronounced Ben-ish

Back to back butterflies

N-alternate paths between any pair

*Is non-blocking*
SHUFFLE/OMEGA NETWORK
(ISOMORPHIC BUTTERFLY)

Shuffle Network

2-ary 3-fly
FLATTENED BUTTERFLY
Clos Networks: (M, N, R)

- **m** = number of middle switches
- **n** = number of input (output) ports on input (output) switches
- **r** = number of input / output switches

3-stages

Clos (5, 3, 4)
A clos network is strictly non-blocking for unicast traffic iff $m \geq 2n-1$

- an unused input on an ingress switch can always be connected to an unused output on an egress switch without having to re-arrange existing routes

**Proof (1953):**

- Suppose an input switch has one free terminal and this has to be connected to a free terminal of an output switch

**Worst case**

- $(n-1)$ input terminals of input switch use $(n-1)$ separate middle switches
- $(n-1)$ output terminals of output switch use $(n-1)$ separate middle switches
- We need another middle switch to connect this input to output
- Total = $(n-1) + (n-1) + 1 = 2n-1$
NON-BLOCKING CLOS

- A clos network is **rearrangeably non-blocking** for unicast traffic iff \( m \geq n \)
  - an unused input on an ingress switch can always be connected to an unused output on an egress switch but this might require *re-arranging of existing routes*

- Proof (1953):
  - If \( m = n \), each input can use one middle switch to connect to its output
**Binary Fat Tree**

- **Diameter?**
- **Bisection Bandwidth?** $N$
- **Diameter?** $2\log_2 N$

*Can be built by folding a multi-stage clos*
Beneš → Folded Clos
Advantages:
- Low diameter
- Fewer links

Disadvantages:
- Lower bisection bandwidth
- Link at concentrator can become bottleneck
MORE HIERARCHICAL TOPOLOGIES

(a) 64 Optically-Connected Clusters
(b) Electrical In-Hub Networks Connecting 16 Cores
(c) core architecture

ANet = ONet + BNet + EMesh

ATAC: PACT 2010
WHICH TOPOLOGY SHOULD YOU CHOOSE?

- Hard to optimize for everything
  - Desired bandwidth
  - Desired latency

- Physical Constraints
  - **Wire budget**
    - Indirect topologies popular off-chip
    - On-chip networks often use direct topologies due to wiring constraints
  - **Wire layout**
    - Topologies should be easy to layout on a planar 2D substrate
  - **Router complexity**
    - Number of ports
MICRO-SWITCH NOC