EECS 570
Lecture 19
Interconnects: Topology

Winter 2022
Prof. Yatin Manerkar

http://www.eecs.umich.edu/courses/eecs570/

Multicore Examples (1)

Sun Niagara

- Niagara 2: 8x9 crossbar (area ~= core)
- Rock: Hierarchical crossbar (5x5 crossbar connecting clusters of 4 cores)
Topics to be covered

- Interfaces
- Topology
- Routing
- Flow Control
- Router Microarchitecture
Types of Topologies
Types of Topologies

• Focus on switched topologies
  □ Alternatives: bus and crossbar

□ Bus
  ○ Connects a set of components to a single shared channel
  ○ Effective broadcast medium

□ Crossbar
  ○ Directly connects $n$ inputs to $m$ outputs without intermediate stages
  ○ Fully connected, single hop network
  ○ Component of routers
Types of Topologies

• **Direct**
  - Each router is associated with a terminal node
  - *All* routers are sources and destinations of traffic

• **Indirect**
  - Routers are distinct from terminal nodes
  - Terminal nodes can source/sink traffic
  - Intermediate nodes switch traffic between terminal nodes

• Most on-chip network use direct topologies
Torus (1)

- K-ary n-cube: $k^n$ network nodes
- n-Dimensional grid with k nodes in each dimension

3-ary 2-cube

2,3,4-ary 3-mesh
Torus (2)

- 1D or 2D torus map well to planar substrate for on-chip

- Topologies in Torus Family
  - Ex: Ring -- k-ary 1-cube

- Edge Symmetric
  - Good for load balancing
  - Removing wrap-around links for mesh loses edge symmetry
    - More traffic concentrated on center channels

- Good path diversity

- Exploit locality for near-neighbor traffic
Torus (3)

• Degree = 2n, 2 channels per dimension
  □ All nodes have same degree

• Total channels = 2nN
  □ N is total number of nodes
Mesh

- A torus with end-around connection removed
- Same max node degree
- Higher demand for central channels
  - Load imbalance
Butterfly

- Indirect network

- K-ary n-fly: $k^n$ network nodes

- Every source-dest pair has the same hop count

- Routing from 000 to 010
  - Dest address used to directly route packet
  - Bit $n$ used to select output port at stage $n$

2-ary 3-fly
2 input switch, 3 stages
Butterfly (2)

- No path diversity  \[ |R_{xy}| = 1 \]
  - Can add extra stages for diversity
  - Increase network diameter

```
R_{xy} = 1
```

Diagram of Butterfly network with nodes labeled 0 through 7.
Butterfly (3)

• Hop Count
  - $\log_k N + 1$
  - Does not exploit locality
    - Hop count same regardless of location

• Switch Degree = $2k$

• Requires long wires to implement
Clos network

- 3-stage networks where all input/output nodes are connected to all middle routers
- Key attribute: path diversity
  - Input node can select any middle router
  - Can enable non-blocking routing algorithms

(5,3,4) Clos network
Fat Tree

- Bandwidth remains constant at each level
- Regular Tree: Bandwidth decreases closer to root
Fat Tree (2)

- Can be constructed from folded Clos
- Provides path diversity
Irregular Topologies
Irregular Topologies

- MPSoC design leverages wide variety of IP blocks
  - Regular topologies may not be appropriate given heterogeneity
  - Customized topology
    - Often more power efficient and deliver better performance
- Customize based on traffic characterization
Irregular Topology Example
Topology Customization

• Merging
  - Start with large number of switches
  - Merge adjacent routers to reduce area and power

• Splitting
  - Large crossbar connecting all nodes
  - Iteratively split into multiple small switches
    - Accommodate design constraints
Topology Implementation
Implementation

• Folding
  - Equalize path lengths
    - Reduces max link length
    - Increases length of other links

![Diagram showing folding and path length equalization](image-url)
Concentration

- Don’t need 1:1 ratio of routers to cores
  - Ex: 4 cores concentrated to 1 router

- Can save area and power

- Increases network complexity
  - Concentrator must implement policy for sharing injection bandwidth
  - During bursty communication
    - Can bottleneck
Implication of Abstract Metrics on Implementation

• Degree: useful proxy for router complexity
  - Increasing ports requires additional buffer queues, requestors to allocators, ports to crossbar

  - All contribute to critical path delay, area and power

  - Link complexity does not correlate with degree
    - Link complexity depends on link width
    - Fixed number of wires, link complexity for 2-port vs 3-port is same
Implications (2)

- Hop Count: useful proxy for overall latency and power

- Does not always correlate with latency
  - Depends heavily on router pipeline and link propagation

- Example:
  - Network A with 2 hops, 5 stage pipeline, 4 cycle link traversal
  - Network B with 3 hops, 1 stage pipeline, 1 cycle link traversal
  - Hop Count says A is better than B
  - But A has 18 cycle latency vs 6 cycle latency for B
Topology Summary

- First network design decision

- Critical impact on network latency and throughput
  - Hop count provides first order approximation of message latency
  - Bottleneck channels determine saturation throughput
Routing
Routing Overview

- Discussion of topologies assumed ideal routing

- In practice...
  - Routing algorithms are not ideal

- Goal: distribute traffic **evenly** among paths
  - Avoid hot spots, contention
  - More balanced $\rightarrow$ closer throughput is to ideal

- Keep complexity in mind
Routing Basics

- Once topology is fixed
- Routing algorithm determines path(s) from source to destination
Routing Example

• Some routing options:
  - Greedy: shortest path
  - Uniform random: randomly pick direction
  - Adaptive: send packet in direction with lowest local channel load

• Which gives best worst-case throughput?
Routing Example (2)

- Consider tornado traffic
  - node $i$ sends to $i+3 \mod 8$
Routing Example (3)

• Greedy:
  - All traffic moves counterclockwise
    - Loads counterclockwise with 3 units of traffic
      - Each node gets 1/3 throughput
    - Clockwise channels are idle

• Random:
  - Clockwise channels become bottleneck
    - Load of 5/2
      - Half of traffic traverses 5 links in clockwise direction
      - Gives throughput of 2/5
Routing Example (4)

- Adaptive:
  - Perfect load balancing (some assumptions about implementation)
  - Sends 5/8 of traffic over 3 links, sends 3/8 over 5 links
    - Channel load is 15/8, throughput of 8/15

- Note: worst case throughput just 1 metric designer might optimize
Routing Algorithm Attributes

• Types
  - Deterministic, Oblivious, Adaptive

• Number of destinations
  - Unicast, Multicast, Broadcast?

• Adaptivity
  - Oblivious or Adaptive? Local or Global knowledge?
  - Minimal or non-minimal?

• Implementation
  - Source or node routing?
  - Table or circuit?
Routing Deadlock

- Each packet is occupying a link and waiting for a link
- Without routing restrictions, a resource cycle can occur
  - Leads to deadlock
Types of Routing Algorithms
Deterministic

- All messages from Source to Destination traverse the same path
- Common example: Dimension Order Routing (DOR)
  - Message traverses network dimension by dimension
  - Aka XY routing

- Cons:
  - Eliminates any path diversity provided by topology
  - Poor load balancing

- Pros:
  - Simple and inexpensive to implement
  - Deadlock-free
Dimension Order Routing

- a.k.a X-Y Routing
  - Traverse network dimension by dimension
  - Can only turn to Y dimension after finished X
Oblivious

- Routing decisions are made without regard to network state
  - Keeps algorithms simple
  - Unable to adapt

- Deterministic algorithms are a subset of oblivious
Valiant’s Routing Algorithm

• To route from $s$ to $d$
  - Randomly choose intermediate node $d'$
  - Route from $s$ to $d'$ and from $d'$ to $d$.

• Randomizes any traffic pattern
  - All patterns appear uniform random
  - Balances network load

• Non-minimal

• Destroys locality
Minimal Oblivious

- Valiant’s: Load balancing but significant increase in hop count
- Minimal Oblivious: some load balancing, but use shortest paths
  - $d'$ must lie within min quadrant
  - 6 options for $d'$
  - Only 3 different paths
Oblivious Routing

- Valiant’s and Minimal Adaptive
  - Deadlock free
    - When used in conjunction with X-Y routing

- Randomly choose between X-Y and Y-X routes
  - Oblivious but not deadlock free!
Adaptive

- Exploits path diversity
- Uses network state to make routing decisions
  - Buffer occupancies often used
  - Coupled with flow control mechanism
- Local information readily available
  - Global information more costly to obtain
  - Network state can change rapidly
  - Use of local information can lead to non-optimal choices
- Can be minimal or non-minimal
Minimal Adaptive Routing

- Local info can result in sub-optimal choices
Non-minimal adaptive

- Fully adaptive
- Not restricted to take shortest path

- Misrouting: directing packet along non-productive channel
  - Priority given to productive output
  - Some algorithms forbid U-turns

- Livelock potential: traversing network without ever reaching destination
  - Mechanism to guarantee forward progress
    - Limit number of misroutings
Non-minimal routing example

- Longer path with potentially lower latency
- Livelock: continue routing in cycle
Adaptive Routing Example

• Should 3 route clockwise or counterclockwise to 7?
  - If 5 is using all the capacity of link 5 → 6...
  - ...queue at node 5 will sense contention but not at node 3

• Backpressure: allows nodes to indirectly sense congestion
  - Queue in one node fills up, it will stop receiving flits
  - Previous queue will fill up

• If each queue holds 4 packets
  - 3 will send 8 packets before sensing congestion
Adaptive Routing: Turn Model

- DOR eliminates 4 turns
  - N to E, N to W, S to E, S to W
  - No adaptivity
- Some adaptivity by removing 2 of 8 turns
  - Remains deadlock free (like DOR)
- West first
  - Eliminates S to W and N to W
**Turn Model Routing**

- **Negative first**
  - Eliminates E to S and N to W

- **North last**
  - Eliminates N to E and N to W

- **Odd-Even**
  - Eliminates 2 turns depending on if current node is in odd or even col.
    - Even column: E to N and N to W
    - Odd column: E to S and S to W
  - Deadlock free if 180 turns are disallowed
  - Better adaptivity
Negative-First Routing Example

- Limited or no adaptivity for certain source-destination pairs
• What about eliminating turns NW and WN?

• Not a valid turn elimination
  - Resource cycle results
Adaptive Routing and Deadlock

• Option 1: Eliminate turns that lead to deadlock
  - Limits flexibility

• Option 2: Allow all turns
  - Give more flexibility
  - Must use other mechanism to prevent deadlock
  - Rely on flow control (later)
    - Escape virtual channels
Routing Algorithm Implementation
Routing Implementation

• Source tables
  □ Entire route specified at source
  □ Avoids per-hop routing latency
  □ Unable to adapt dynamically to network conditions
  □ Can specify multiple routes per destination
    ☐ Give fault tolerance and load balance
  □ Support reconfiguration
### Source Table Routing

<table>
<thead>
<tr>
<th>Destination</th>
<th>Route 1</th>
<th>Route 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>10</td>
<td>EX</td>
<td>EX</td>
</tr>
<tr>
<td>20</td>
<td>EEX</td>
<td>EEX</td>
</tr>
<tr>
<td>01</td>
<td>NX</td>
<td>NX</td>
</tr>
<tr>
<td>11</td>
<td>NEX</td>
<td>ENX</td>
</tr>
<tr>
<td>21</td>
<td>NEEX</td>
<td>ENEX</td>
</tr>
<tr>
<td>02</td>
<td>NNX</td>
<td>NNX</td>
</tr>
<tr>
<td>12</td>
<td>ENNX</td>
<td>NNEX</td>
</tr>
<tr>
<td>22</td>
<td>EENNX</td>
<td>NNEEX</td>
</tr>
<tr>
<td>03</td>
<td>NNNX</td>
<td>NNNX</td>
</tr>
<tr>
<td>13</td>
<td>NENNIX</td>
<td>ENNNX</td>
</tr>
<tr>
<td>23</td>
<td>EENNNX</td>
<td>NNNEEX</td>
</tr>
</tbody>
</table>

- Arbitrary length paths: storage overhead and packet overhead

![Network Diagram](attachment:image)
Node Tables

• Store only next direction at each node

• Smaller tables than source routing

• Adds per-hop routing latency

• Can adapt to network conditions
  □ Specify multiple possible outputs per destination
  □ Select randomly to improve load balancing
## Node Table Routing

<table>
<thead>
<tr>
<th>From</th>
<th>00</th>
<th>01</th>
<th>02</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>20</th>
<th>21</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>X</td>
<td>N</td>
<td>N</td>
<td>E</td>
<td>E</td>
<td>N</td>
<td>E</td>
<td>N</td>
<td>E</td>
</tr>
<tr>
<td>01</td>
<td>S</td>
<td>X</td>
<td>N</td>
<td>E</td>
<td>S</td>
<td>E</td>
<td>E</td>
<td>N</td>
<td>E</td>
</tr>
<tr>
<td>02</td>
<td>S</td>
<td>S</td>
<td>X</td>
<td>E</td>
<td>S</td>
<td>E</td>
<td>E</td>
<td>S</td>
<td>E</td>
</tr>
<tr>
<td>10</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>X</td>
<td>N</td>
<td>N</td>
<td>E</td>
<td>E</td>
<td>N</td>
</tr>
<tr>
<td>11</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>S</td>
<td>X</td>
<td>N</td>
<td>E</td>
<td>S</td>
<td>E</td>
</tr>
<tr>
<td>12</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>S</td>
<td>S</td>
<td>X</td>
<td>E</td>
<td>S</td>
<td>E</td>
</tr>
<tr>
<td>20</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>X</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>21</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>S</td>
<td>X</td>
<td>N</td>
</tr>
<tr>
<td>22</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>S</td>
<td>S</td>
<td>X</td>
</tr>
</tbody>
</table>

- Each node would have 1 row of table
  - Max two possible output ports
Implementation

• Combinational circuits can be used
  □ Simple (e.g. DOR): low router overhead
  □ Specific to one topology and one routing algorithm
    ○ Limits fault tolerance

• OTOH, tables can be updated to reflect new configuration, network faults, etc
Circuit Based

- Next hop based on buffer occupancies in a 2D mesh
- Or could implement simple DOR
- Fixed w.r.t. topology

Productive Direction Vector
Queue lengths
Selected Direction Vector

Route selection
## Routing Algorithms: Implementation

<table>
<thead>
<tr>
<th>Routing Algorithm</th>
<th>Source Routing</th>
<th>Combinational</th>
<th>Node Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOR</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Oblivious</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Valiant’s</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Minimal</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Adaptive</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Routing: Irregular Topologies

• MPSoCs
  ☐ Power and performance benefits from irregular/custom topologies

• Common routing implementations
  ☐ Rely on source or node table routing

• Maintain deadlock freedom
  ☐ Turn model may not be feasible
    ☐ Limited connectivity
Routing Summary

- Latency paramount concern
  - Minimal routing most common for NoC
  - Non-minimal can avoid congestion and deliver low latency

- To date: NoC research favors DOR for simplicity and deadlock freedom
  - On-chip networks often lightly loaded

- Only covered unicast routing
  - Recent work on extending on-chip routing to support multicast