Introduction to Scalable Interconnection Networks

COE 502 – Parallel Processing Architectures
Prof. Muhamed Mudawar
Computer Engineering Department
King Fahd University of Petroleum and Minerals

Scalable Interconnection Network

- At Core of Parallel Computer Architecture
- Transfer data from any source to any destination
- Composed of links and switches
  - Elegant mathematical structure (highly regular)
  - Electrical / Optical link properties
  - Managing many traffic flows
- Performance Goals
  - Bandwidth
    - As many concurrent transfers as possible
  - Latency: as small as possible
  - Cost: as low as possible
Formalism

- Interconnection Network is a graph
- Vertices $V = \{\text{nodes, switches}\}$
- Connected by communication channels $C \subseteq V \times V$
- A Channel is a physical link
  - Includes buffers to hold data as it is being transferred
  - Phit (Physical unit) is amount of data transferred per cycle
  - $\tau$ is the channel cycle: time to transmit one phit
  - Channel has signaling rate $f = 1/\tau$
  - Channel has width $w$ and bandwidth $b = w \times f$
- Switch Degree: number of input (output) channels
- Path or Route: sequence of switches and links
  - Followed by a message from its source until its destination

Network Characterization

- Topology (what structure)
  - Physical interconnection structure of the network graph
  - Direct: a switch is associated with each node
  - Indirect: can have extra switches not connected to nodes
  - Regular versus Irregular
  - Most parallel machines employ highly regular topologies
- Routing Algorithm (which routes)
  - Restricts the set of paths that messages may follow
    - Between pairs of source and destination nodes
  - Deterministic versus adaptive
    - One or multiple routes for each pair of source/destination
  - Many algorithms with different properties
Network Characterization (2)

- **Switching Strategy** (how)
  - How data in a message traverses a route
  - Circuit switching versus packet switching
    - In circuit switching, path is established and reserved
      - Until message traverses over circuit
    - In packet switching, message is broken into packets
      - Packets contain routing/sequencing information, and data

- **Flow Control Mechanism** (when)
  - When a message or portions of it traverse a route
  - What happens when messages compete for a channel?
    - Blocked in place, buffered, detoured, dropped
  - **Flow control unit (Flit):** unit of transfer across a link
    - Can be as small as a phit or as large as a packet

Typical Packet Format

- **Header**
  - Front end of the packet
  - Routing and control info
  - Used by switches to route packet in network

- **Data payload:** data transmitted across network

- **Trailer:** end of packet
  - Typically contains error-checking code

- **Packet is further divided into flits and phits**

- **Example: Cray T3E**
  - Packet is 1-10 flits, and each flit is 5 phits
  - Flit size = 70 bits = 64-bit data + 6-bit control
Switch Consists of:
- Set of input ports and output ports
- Internal crossbar connecting each input to every output
- Internal buffering
- Control logic for routing and scheduling

Output ports
- Transmitter: typically drives clock and data

Input ports
- Receiver aligns data signal with local clock
- Essentially FIFO buffer

Buffering at input and/or output ports

Crossbar
- Connects each input to any output
- Switch degree limited by number of I/O pins

Control logic
- Complexity depends on routing and scheduling algorithm
- Determines output port for each incoming packet
- Arbitrates among inputs directed to same output
Physical Channel Flow Control

- Asynchronous physical channel flow control

- Synchronous full-duplex channel flow control

Topological Properties

- Routing Distance
  - Number of links on route between a pair of nodes

- Network Diameter
  - Maximum shortest path between any two nodes

- Average Distance
  - Average of the routing distance between all pairs of nodes

- Channel Bisection Width
  - Minimum number of channels cut
    - When a network is cut into two equal halves

- Wire Bisection Width
  - Channel bisection width × channel width
  - Reflects the wiring density of the network
Interconnection Topologies

- Each topology is a class of networks
  - Scaling with number of nodes $N$

- Completely connected network
  - Each node has a switch
  - Directly connected to all other nodes
  - Node Degree = $N - 1$
  - Diameter = 1 link
  - Links = $N (N - 1) / 2$
  - Bisection width = $(N/2)^2$
    - Each of the $(N/2)$ nodes in the first half is connected to all the $(N/2)$ nodes in the second half

Linear Array

- Switch associated with each node
- Connected by bidirectional links
- Number of links = $N - 1$
- Diameter = $N - 1$
- Average distance = $(N+1)/3$
- Node Degree = 2
- Bisection width = 1 link
  - Removal of a single link partitions the network
- One route between a pair of nodes
  - Route $A \rightarrow B$ is given by relative address $R = B - A$
**Ring**

- Symmetric, Number of links = \( N \)
- Bidirectional Links
  - Diameter = \( N / 2 \)
  - Node Degree = 2
  - Average distance = \( N^2 / 4(N - 1) \)
  - Bisection width = 2 links
  - Two routes between a pair of nodes
- Unidirectional Links
  - Diameter = \( N - 1 \)
  - Node Degree = 1
  - Average distance = \( N / 2 \)
  - Bisection width = 1 link
  - One route between a pair of nodes

---

**Multidimensional Meshes**

- \( d \)-dimensional array
  - \( N = k_0 \times \ldots \times k_{d-1} \) nodes
  - \( k_i \) nodes in dimension \( i \)
  - Node degree is between \( d \) and \( 2d \)
  - Each node identified by \( d \)-vector of coordinates \((x_0, \ldots, x_{d-1})\)
    - \( 0 \leq x_i \leq k_i - 1 \) for \( 0 \leq i \leq d - 1 \)
  - If number of nodes is same \((k)\) in all dimensions …
    - Then \( d \)-dimensional \( k \)-ary mesh
    - \( N = k^d \)
    - Network diameter = \( d(k-1) \)
    - Bisection width = \( k^{d-1} \)
**Multidimensional Tori**

- Symmetric with wrap around edges
- Node degree = $2d$
- $N = k_0 \times \ldots \times k_{d-1}$ nodes
- $k_i$ nodes in dimension $i$
  - Each node identified by $d$-vector of coordinates $(x_0, \ldots, x_{d-1})$
  - Where $0 \leq x_i \leq k_i - 1$ for $0 \leq i \leq d - 1$
- If number of nodes is same ($k$) in all dimensions …
  - Then $d$-dimensional $k$-ary torus
  - $N = k^d$
  - Network diameter = $d[k/2]$
  - Number of links = $dN$
  - Bisection width = $2k^{d-1}$

**Hypercube**

- Special case of $d$-dimensional $k$-ary mesh
- Called also $d$-cube
- $d$ dimensions
- Two nodes along each dimension
- Node degree = $d$
- $N = 2^d$ nodes
- Network diameter = $d$
- Number of links = $dN / 2$
- Bisection width = $N / 2$
**Latency**

- Time to transfer \( n \) bytes from source to destination
  
  \( \text{Overhead} + \text{Unloaded Network Latency} + \text{Contention Delay} \)

- Overhead
  
  - Time to get message into and out of network
  
  - Node-to-network interface

- Unloaded Network Latency
  
  - Time to transfer a packet through network
  
  - Assuming no contention in the network
  
  - Further divided into: channel occupancy + routing delay

- Contention Delay
  
  - Contention adds queuing delays (waiting time in buffers)

---

**Store-and-Forward Routing**

- Entire packet is received at a switch and then …
  
  - Forwarded on the next link along the path

\[
\begin{align*}
\text{Source} & \rightarrow & \text{Dest} \\
3 & 2 & 1 & 0 & \rightarrow & 0 \\
3 & 2 & 1 & 3 & \rightarrow & 3 \\
3 & 2 & 1 & 0 & \rightarrow & 0 \\
3 & 2 & 1 & 0 & \rightarrow & 3 \\
3 & 2 & 1 & 0 & \rightarrow & 0 \\
3 & 2 & 1 & 0 & \rightarrow & 3 \\
3 & 2 & 1 & 0 & \rightarrow & 3 \\
3 & 2 & 1 & 0 & \rightarrow & 3 \\
3 & 2 & 1 & 0 & \rightarrow & 3 \\
\end{align*}
\]

4-flit packet traverses 3 hops from source to destination

Packet = \( n \) bytes
Routing distance = \( h \)
Additional Switch Delay = \( \Delta \)
Link bandwidth = \( b \) bytes/sec

Unloaded network latency:
\[
T_{SF} (n, h) = h \left( \frac{n}{b} + \Delta \right)
\]
**Cut-through Routing**

- Transmission of a single packet is pipelined
- Switch makes it decision after examining header flit
  - Advances header before receiving remaining flits
- Header establishes route from source to destination
  - A single packet may occupy entire route
  - Tail (last) flit clears route as it moves through

Packet = \( n \) bytes  
Routing distance = \( h \)  
Routing delay per hop = \( \Delta \)  
Link bandwidth = \( b \) bytes/s  

Unloaded network latency:  
\[ T_{CT}(n, h) = \frac{n}{b} + h\Delta \]  

---

**Channel Occupancy**

- Time for a packet to cross a channel  
- Channel Occupancy = \( \frac{n}{b} = \frac{n_D + n_E}{b} \)  
  - Packet = \( n \) bytes = \( n_D + n_E \) (data + envelop)  
  - Packet envelop include the header and trailer flits
    - Typically discarded when a packet reaches its destination  
    - Counted as an overhead (routing info, error codes, etc.)  
  - Packet efficiency = \( \frac{n_D}{n_D + n_E} \)  
  - Channel bandwidth \( b = \frac{w}{\tau} \)  

- Channel Occupancy for store-and-forward = \( h \times \frac{n}{b} \)  
  - Not overlapped along route, multiplied by distance \( h \)  
- Channel Occupancy for cut-through routing = \( \frac{n}{b} \)  
  - Overlapped in time and does not depend on distance \( h \)
Routing Delay

- Time to route header flit from source to destination
- Is a function of
  - Routing distance \( h \) and
  - Routing delay \( \Delta \) incurred at each hop along the path

Routing Delay = \( h \Delta \)
- For both store-and-forward and cut-through routing

\( \Delta \) is the routing delay per hop, which includes
- Routing logic delay to determine output port for a header flit
- Crossbar delay to advance header flit from input to output
- Once a path has been established for a header flit
  - All remaining flits will simply follow with no additional delay

Real Machines

<table>
<thead>
<tr>
<th>Machine</th>
<th>Topology</th>
<th>Cycle Time (ns)</th>
<th>Channel Width (bits)</th>
<th>Routing Delay (cycles)</th>
<th>Flit (bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>nCube/2</td>
<td>Hypercube</td>
<td>25</td>
<td>1</td>
<td>40</td>
<td>32</td>
</tr>
<tr>
<td>TMC CM-5</td>
<td>Fat-Tree</td>
<td>25</td>
<td>4</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>IBM SP-2</td>
<td>Banyan</td>
<td>25</td>
<td>8</td>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>Intel Paragon</td>
<td>2D Mesh</td>
<td>11.5</td>
<td>16</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>Meiko CS-2</td>
<td>Fat-Tree</td>
<td>20</td>
<td>8</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>CRAY T3D</td>
<td>3D Torus</td>
<td>6.67</td>
<td>16</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>DASH</td>
<td>Torus</td>
<td>30</td>
<td>16</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>J-Machine</td>
<td>3D Mesh</td>
<td>31</td>
<td>8</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>Monsoon</td>
<td>Butterfly</td>
<td>20</td>
<td>16</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>SGI Origin</td>
<td>Hypercube</td>
<td>2.5</td>
<td>20</td>
<td>16</td>
<td>160</td>
</tr>
<tr>
<td>Myricom</td>
<td>Arbitrary</td>
<td>6.25</td>
<td>16</td>
<td>50</td>
<td>16</td>
</tr>
</tbody>
</table>
Contention

- Two packets trying to use same link at same time
  - Depends on topology, destination, and routing algorithm
- Contention adds queuing delay to basic routing delay
- Mechanism for dealing with contention
  - Means of buffering
    - Buffer entire packet
    - Buffer few flits of a packet
  - What happens when buffer is full?
    - Discard packet
    - Back pressure toward the source
  - Means of arbitration for the output channels

Mechanisms for Contention

- Store-and-forward
  - Entire packet is blocked in buffer until arbiter selects it
  - What happens to incoming packets when buffer is full?
    - Handshake between output and input port across a link
    - Packet heading to a full buffer is blocked in place
    - Discarded in traditional networks because of long links
- Cut-through: two mechanisms exist for contention
  - Virtual Cut-through
    - Buffer space is large enough to store the entire blocked packet
    - Frees previous buffers along the route
  - Wormhole
    - Buffer space can hold one of few flits of a packet
    - Packet is blocked in all buffers along its route
    - Eventually the source experiences back pressure
Routing

- Routing algorithm determines
  - Which of the possible paths are used as routes
  - Routing algorithm is a function $R : V \times V \rightarrow C$
  - At each switch $\in V$, routing function maps
    - Destination node $\in V$ to next channel $\in C$ on route

Routing mechanisms

- Simple Arithmetic: minimal computation in few cycles
  - Works in most regular topologies
- Source-Based Routing
  - Source builds a header consisting of the output port numbers
  - Each switch simply removes one port number from header flit
- Routing Table $R$
  - Header contains a routing field $i$, output port $o = R[i]$
  - Routing table also gives the routing field for next step $j = R[i]$

Routing Mechanisms - cont’d

- Source-based
  - Routing algorithm is applied at source node, not in switches
  - Source node computes a series of output port selects
  - Ports are carried in message header
  - Used by switches and stripped en route
  - Very simple switch design but header tends to be large
  - Examples: CS-2, Myrinet, MIT Artic

- Table-driven
  - Message header carries routing index for next switch
  - Routing table is indexed to obtain output port and next index
    - $(o, j) = R[i]$, where $o = $ output port and $j = $ next index
  - Example: ATM - Not common in interconnection networks
  - Fairly large tables even for simple routing algorithms
Deterministic Routing

- Unique path between every source and destination
- Dimension-Order Routing (DOR) in 2D Mesh
  - Each packet carries a signed distance $[\Delta x, \Delta y]$ in its header
  - Route along $X$ dimension first, then along $Y$ dimension

<table>
<thead>
<tr>
<th>Condition</th>
<th>Direction (Output port) and Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta x &lt; 0$</td>
<td>West (-$X$), Increment $\Delta x$</td>
</tr>
<tr>
<td>$\Delta x &gt; 0$</td>
<td>East (+$X$), Decrement $\Delta x$</td>
</tr>
<tr>
<td>$\Delta x = 0$, $\Delta y &lt; 0$</td>
<td>South (-$Y$), Increment $\Delta y$</td>
</tr>
<tr>
<td>$\Delta x = 0$, $\Delta y &gt; 0$</td>
<td>North (+$Y$), Decrement $\Delta y$</td>
</tr>
<tr>
<td>$\Delta x = 0$, $\Delta y = 0$</td>
<td>Processor</td>
</tr>
</tbody>
</table>

- Can be generalized to $k$-ary $d$-dimensional meshes and tori
- Similar e-cube routing in $d$-dimensional hypercube
  - One routing bit per dimension

DOR and E-Cube Routing Examples

- Examples on Dimension Order Routing (DOR)

- Examples on e-Cube Routing
**Adaptive Routing**

- Multiple paths may exit between source & destination
- **Routing algorithm** determines multiple output ports
  - For an incoming packet based on destination address
- **Selection function** is used to select an output port
  - Based on traffic and contention to output ports
- **Minimal adaptive routing**
  - Minimal paths are chosen between source & destination
- Example showing 5 minimal paths between 2 nodes

**Deadlock**

- How can it arise?
  - Necessary conditions:
    - Shared resources
    - Channels and buffers
    - Incrementally allocated
      - When header flit arrives
    - No preemption
      - Remain allocated until last flit
    - Cyclic dependencies
      - Messages are waiting on each other in a cyclic manner
- How to prevent deadlock?
  - Break cyclic dependencies by
    - Constraining resource allocation

4 messages waiting on each other in a cyclic manner
Deadlock-Free Routing

- Deadlocks are a disaster for a parallel machine
  - Once a deadlock happens, no progress can take place
  - Until machine is restarted and buffers are reset and cleared
- Packets introduce dependences between channels
  - As they move forward between source and destination
- Channel Dependence Graph
  - Describes dependences between channels
  - For a given topology and routing algorithm
  - Has a node for every unidirectional link in the network
  - Arc from node \( a \) to node \( b \) if …
  - It is possible for a packet to traverse from channel \( a \) to \( b \)
- No cycles in graph \( \Rightarrow \) Deadlock-free routing

DOR in 2D Mesh

- To Prove: DOR in 2D Mesh is Deadlock Free
- Assign Channel Numbers
  - Such that every legal route follows an ordered sequence
  - Either monotonically increasing or decreasing
- In this example, \( k = 4 \) and \( N = 16 \)
  - Channel Numbering
  - \(+X: (x, y) \rightarrow (x+1, y)\) gets \( 2k y + x \)
  - \(-X: (x, y) \rightarrow (x-1, y)\) gets \( 2k (y + 1) - x \)
  - \(+Y: (x, y) \rightarrow (x, y+1)\) gets \( 2(N + k x) + y \)
  - \(-Y: (x, y) \rightarrow (x, y-1)\) gets \( 2(N + k x + k) - y \)
  - Any routing sequence: \( X \) turn \( Y \)
    - Always increasing
Channel Dependence Graph

- Channel dependency graph shows all possible routes for DOR in a 2D Mesh network.
- No cycles => DOR in 2D mesh is deadlock free.