ECE 259 / CPS 221
Advanced Computer Architecture II
(Parallel Computer Architecture)

Interconnection Networks

Copyright 2006 Daniel J. Sorin
Duke University

Slides are derived from work by
Sarita Adve (Illinois), Babak Falsafi (CMU),
Mark Hill (Wisconsin), Alvy Lebeck (Duke), Steve
Reinhardt (Michigan), and J. P. Singh (Princeton).
Thanks!

Interconnection Networks

- **Goal**: Communication between computers
  - Try to achieve low latency and high bandwidth
- **Warning**: Terminology-rich environment
- We'll focus on networks for parallel computing
  - Today's System Area Networks exhibit many of the same properties
  - Many concepts similar to general networking
    - » But the parameters can be very different!

Some Terminology

Network characterized by

- **Topology**
  - Physical structure of the graph
- **Routing Algorithm**
  - What paths through network can a message follow
- **Switching Strategy**
  - How data in message traverses its route
  - Circuit Switched vs Packet Switched
- **Flow Control**
  - When does a packet (or portions of it) move along its route

More Terminology

- Given a topology constructed by linking switches and network interfaces, we must deliver a packet from node A to node B
- **Link**: cable with connectors on each end
  - Connects switches to other switches or network interfaces
- **Switch**: connect N inputs to N outputs (degree N)
- **Phit**: Minimum # of bits physically moved across link in one cycle (can pipeline on single wire)
- **Flit**: Minimum # of bits move across link as single unit (for purposes of flow control)
- **Packet**: Unit that requires routing information, some number of flits

Outline

- Topology
- Switching, Routing, & Deadlock
- Switch Design
- Flow Control
- Case Studies

Topology

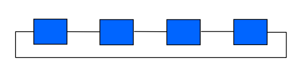
- Topology is the structure of the interconnect
  - Geometric property → topology has nice mathematical properties
- Topology determines
  - Switch Degree: number of outgoing links from a switch
  - Diameter: number of links crossed between nodes on maximum shortest path
  - Average distance: number of hops to random destination
  - Bisection: minimum number of links that, if removed, would separate the network into two halves
- Direct vs Indirect Networks
  - Direct: All switches attached to host nodes (e.g., mesh)
  - Indirect: Many switches not attached to host nodes (e.g., tree)

k-ary d-cubes

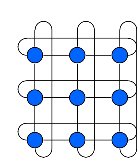
- Often called k-ary n-cubes
- General class of regular, direct topologies
  - Subsumes rings, tori, cubes, etc.
- d dimensions (where d also equal switch degree)
  - 1 for ring
  - 2 for mesh or torus
  - 3 for cube
  - Can choose arbitrarily large d, except for cost of switches
- k switches in each dimension

Examples of k-ary d-cubes

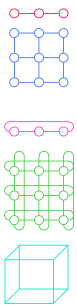
- 1D Ring = k-ary 1-cube
  - d = 1 (always)
  - k = N (always) = 4 (here)
  - Degree = 3 = 2 neighbors + host node (always)
  - Diameter = ? Ave dist = ?
  - Bisection = ?



- 2D Torus = k-ary 2-cube
  - d = 2 (always)
  - k = $\log_d N$ (always) = 3 (here)
  - Degree = 5 = 4 neighbors + host node (always)
  - Diameter = ? Ave dist = ?
  - Bisection = ?



Important k-ary d-cube Topologies



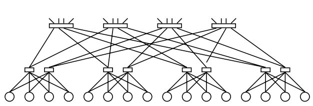
Type	Degree	Diameter	Ave Dist	Bisection	N = 1024 Diam	Ave D
1D mesh	2	N-1	2N/3	1		
2D mesh	4	$2(N^{1/2} - 1)$	$2N^{1/2} / 3$	$N^{1/2}$	63	21
3D mesh	6	$3(N^{1/3} - 1)$	$3N^{1/3} / 3$	$N^{2/3}$	~30	~10
dD mesh	2d	$d(N^{1/d} - 1)$	$dN^{1/d} / 3$	$N^{(d-1) / d}$		
$(N = k^d)$						
Ring	2	N / 2	N/4	2		
2D torus	4	$N^{1/2}$	$N^{1/2} / 2$	$2N^{1/2}$	32	16
k-ary d-cube	2d	$d(N^{1/n})$	$dN^{1/d}/2$		15	8 (3D)
$(N = k^d)$		dk/2	dk/4	$2k^{n-1}$		
Hypercube(k=2)	d	d= LogN	d/2	N/2	10	5

Trees and Tree-Like Topologies

- Indirect topology – most switches not attached to nodes
- Tree: send message up from leaf to closest common ancestor, then down to recipient
- N host nodes at leaves
- k = branching factor of tree (k=2 → binary tree)
  - Switch degree = k+1 (k down links and one uplink)
- d = dimension = height of tree = $\log_k N$
- Diameter = $2\log_k N$ (up and then down)
- Problem with trees: too much contention at or near root
- Fat tree: same as tree, but with more bandwidth near the root (by adding multiple roots and high order switches)

Important Tree-Like Topologies

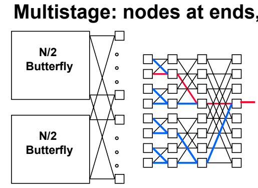
Type	Degree	Diameter	Ave Dist	Bisection	N = 1024 Diam	Ave D
binary tree	3	$2\text{Log}_2 N$	$\sim 2\text{Log}_2 N$	1	20	~20
4-ary tree	5	$2\text{Log}_4 N$	$\sim 2\text{Log}_4 N$	1	10	9.33
k-ary tree	k+1	$\text{Log}_k N$				
binary fat tree	4	$\text{Log}_2 N$		N		
binary butterfly	4	$\text{Log}_2 N$	$\text{Log}_2 N$	N/2	20	20



CM-5 “Thinned” Fat Tree

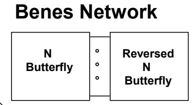
Butterfly

Multistage: nodes at ends, switches in middle



- All paths equal length
- Unique path from any input to any output
- Conflicts cause tree saturation

Benes Network



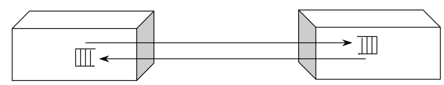
- Routes all permutations w/o conflict
- Notice similarity to fat tree (fold in half)
- Randomization is major breakthrough

Outline

- Topology
- Switching, Routing, & Deadlock
- Switch Design
- Flow Control
- Case Studies

ABCs of Networks

- **Starting Point**: Send bits between 2 computers

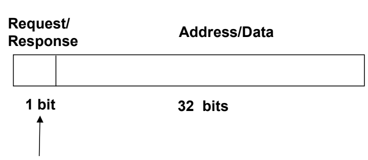


Queue on each end

- Can send both ways ("Bi-directional, full duplex")
- Rules for communication? Protocol
  - Synchronous send
    - » Need request & response signaling
  - Name for standard group of bits sent: Packet

A Simple Example

- What is the packet format?
  - Fixed? (for ease of hardware interpretation)
  - Variable? (for flexibility)



0: Please send data from Address
1: Packet contains data corresponding to request

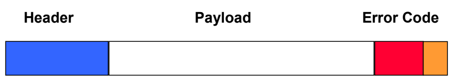
Questions About Simple Example

- What if more than 2 computers want to communicate?
  - Need node identifier field (destination) in packet
  - Routing and topology
- What if packet is garbled in transit?
  - Add error detection field in packet (e.g., CRC)
- What if packet is lost?
  - More elaborate protocols to detect loss (e.g., NAK, time outs)
- What if multiple processes/machine?
  - Queue per process

These issues → more complex protocols & packet formats

General Packet Format

- Header
  - Routing and control information
- Payload
  - Carries data (non hardware-specific information)
  - Can be further divided (framing, protocol stacks…)
- Error code (detecting or correcting)
  - Generally at tail of packet so it can be generated on the way out



Message vs. Packet

- A message may be composed of several packets

- Applications reason about messages
- Networks transfers packets

- Small fixed-size packets
  - Easy for network hardware
  - But can lead to fragmentation and reassembly (SW overhead)

- Variable-size packets
  - Can avoid some fragmentation
  - But can cause congestion and can be tougher for hardware

Packet Switched vs Circuit Switched

Circuit Switched

- Establish route then send data
- Like the telephone system

Packet Switched

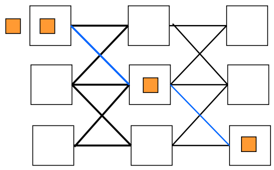
- Route each packet individually
- May make delivery guarantees such as
  - Reliable delivery
  - Point-to-point ordering of packets

Packet Routing

- There are two basic approaches to routing packets, based on what a switch does with a packet as its flits begin to arrive

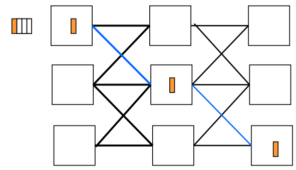
1) Store-and-forward
2) Cut-through
   - Virtual cut-through
   - Wormhole

Store and Forward



- **Store-and-forward** policy: each switch waits for the full packet to arrive in the switch before it is sent on to the next switch

Cut Through



- **Cut-through** routing: switch examines the header, decides where to send the message, and then starts forwarding it immediately
- Two flavors of cut-through based on what switch does if output port is blocked …

Virtual Cut-Through

- What to do if output port is blocked?
- Allow the tail to continue when the head is blocked, absorbing the whole message into a single switch
  - Requires a buffer large enough to hold the largest packet
- Degenerates to store-and-forward with high contention

Wormhole

- When the head of the message is blocked, the message stays strung out over the network
  - Potentially blocks other messages (needs only buffer the piece of the packet that is sent between switches).
  - CM-5 used it, with each switch buffer being 4 bits per port
  - Myrinet uses it
- Can cause tree saturation

Store and Forward vs. Cut-Through

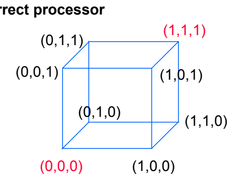
- Store and Forward latency is function of
  - Number of intermediate switches times the size of the packet
- Cut-through latency is function of
  - time for 1st part of packet to negotiate the switches + (packet size ÷ interconnect bandwidth)
- Which seems better?

Routing Algorithm

- How do I know where a packet should go?
  - Topology does NOT determine routing (e.g., many paths thru torus)
- Many routing algorithms exist
  1) Arithmetic
  2) Source-based
  3) Table lookup
  4) Adaptive—route based on network state (e.g., contention)

(1) Arithmetic Routing

- For regular topology, use simple arithmetic to determine route
- E.g., 3D Torus
  - Packet header contains signed offset to destination (per dimension)
  - At each hop, switch +/- to reduce offset in a dimension
  - When x == 0 and y == 0, then at correct processor



- Drawbacks
  - Requires ALU in switch
  - Must re-compute CRC at each hop

(2) Source Based & (3) Table Lookup Routing

Source Based

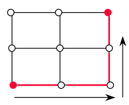
- Source specifies output port for each switch in route
- Very simple switches
  - No control state
  - Strip output port off header
- Myrinet uses this
- Can't be made adaptive

Table Lookup

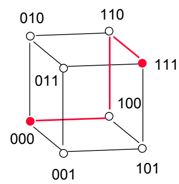
- Very small header, index into table for output port
- Big tables, must be kept up-to-date

Deterministic vs. Adaptive Routing

- **Deterministic—follows a pre-specified route**
  - K-ary d-cube: dimension-order routing
    - » (x1, y1) → (x2, y2)
    - » First Dx = x2 - x1,
    - » Then Dy = y2 - y1,
  - Tree: common ancestor

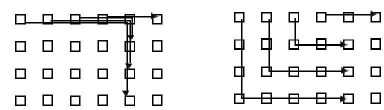


- **Adaptive—route determined by contention for output port**



(4) Adaptive Routing

- Essential for fault tolerance
  - At least multipath
- Can improve utilization of the network
- Simple deterministic algorithms easily run into bad permutations



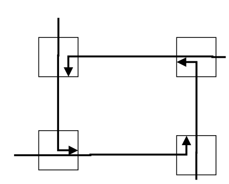
- Fully/partially adaptive, minimal/non-minimal
- Can introduce complexity or anomalies
- A little adaptation goes a long way!

Hot Potato Routing

- Every cycle, each switch takes each input and routes it to an output
  - But not necessarily to the "desired" output
- No switch buffering!
- Possibility of livelock if no precautions taken
  - E.g., could grant priority based on age of packet

Deadlock

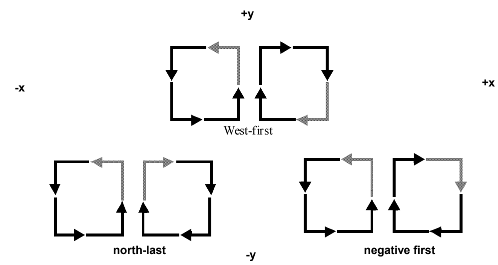
- Necessary conditions to achieve deadlock
  - Use more than one resource
  - Not willing to release resource in use
  - Cycle in order of recourse use



Deadlock Free Routing

- Virtual Channels
  - Not to be confused with "virtual cut-through"
  - Add buffers so flits of wormhole packets can be interleaved
  - We'll read about this in Dally's paper
- Up*-Down*
  - Number switches: higher = farther away from processors
  - Route up, make one turn, route down
- Turn Model Routing
  - Restrict order of turns
    - » West first
    - » North last
    - » Negative first
  - Can increase number of hops

Minimal turn restrictions in 2D

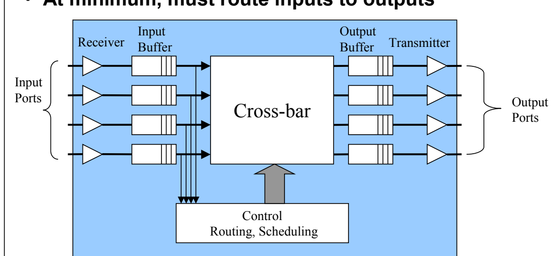


Outline

- Topology
- Switching, Routing, & Deadlock
- Switch Design
- Flow Control
- Case Studies

A Generic Switch

- At minimum, must route inputs to outputs



VLSI makes it easier to create larger fully connected switches

Switch Design Issues

- Ports
  - How many ports can we have before pins become the limit?
- Datapath (internal crossbar design)
  - Can we design a non-blocking crossbar?
- Routing logic per input
  - ALU
  - Table
  - Finite State Machine (FSM) for cut-through

Switch Buffering

- Must absorb burstiness in traffic
  - Unless using hot potato routing
- Options
  - Shared, centralized buffer
  - Input buffering
  - Output buffering
- Shared buffer pool
  - Need high bandwidth
  - One congested output port could hog all buffer space

Input Buffering

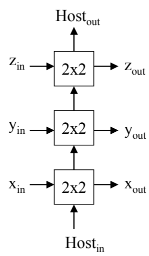
- Buffer per input port
- Routing logic associated with each input port
- Problem: Head of line (HOL) blocking
  - Subsequent packet may be routed to unused output port

Output Buffering

- Buffers logically associated with output
  - Split on either side of crossbar
- Arbitration for physical link (output scheduling)
  - Static priority
  - Random
  - Round-robin
  - Oldest-first
- Effects of adaptive routing?
  - Select output based on availability
  - Requires feedback from output port

Stacked Dimension Switch

- Uses only 2x2 switch to build higher dimension switch (2x2 switches have simpler designs)



Outline

- Topology
- Switching, Routing, & Deadlock
- Switch Design
- Flow Control
- Case Studies

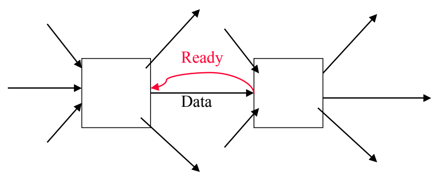
Congestion Control

- Packet switched networks do not reserve bandwidth, which can lead to contention
- Solution: prevent packets from entering until contention is reduced (e.g., metering lights)
- Options:
  - End-to-end flow control
  - Link-level flow control

Flow Control

- Packet discarding: If a packet arrives at a switch and there is no room in the buffer, the packet is discarded
  - No communication between switches, requires higher level protocol
- Flow control: between pairs of receivers and senders; use feedback to tell the sender when it is allowed to send the next packet
  - Link-level: flow control done on per-link basis
  - End-to-end: flow control done over entire path length

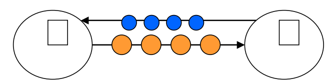
Link-Level Flow Control



- Transfer single flit when receiver is ready
- Could have long links with many flits in flight

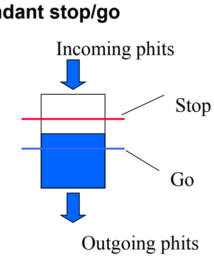
Credit-based (Window) Flow Control

- **Receiver gives N credits to sender**
  - **Sender decrements count**
  - **Stops sending if zero**
  - **Receiver sends back credit as it drains its buffer**
  - **Bundle credits to reduce overhead**
- **Must account for link latency**



Water Level

- **High water, low water**
- **Stop & go back to source switch (Myrinet)**
- **Can send redundant stop/go**



Virtual Channel Flow Control

- **PRESENTATION**

Outline

- Topology
- Switching, Routing, & Deadlock
- Switch Design
- Flow Control
- Case Studies

Case Study Cray T3D

- 1024 switch nodes each connected to 2 processors
- 3D torus, bidirectional, 300 MB/s
- Link: 16 bits, 8 control bits
- Variable size packet (multiple of 16 bits)
- Logical request & response networks
  - 2 virtual channels each for deadlock
- Stacked dimension routing
- Wormhole for large packets, virtual cut-through for small packets

Real (But Old) Machines

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

Alpha 21364 (EV7) Network

- PRESENTATION