CSCI-1680
Switching

Rodrigo Fonseca

Based partly on lecture notes by David Mazières, Phil Levis, John Jannotti

Administrivia

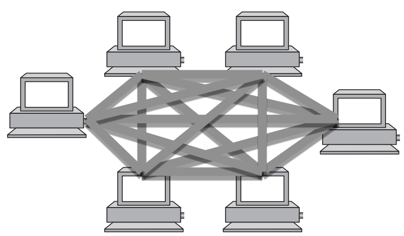
- Homework I out, due next Friday, Feb 18
- No class next Tuesday

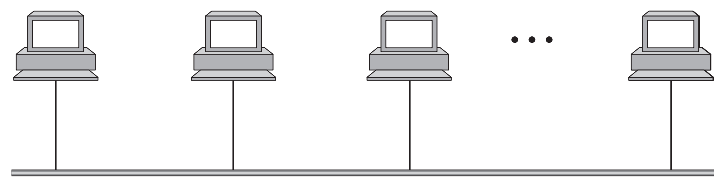
Today

- Ethernet (cont.)
- Link Layer Switching

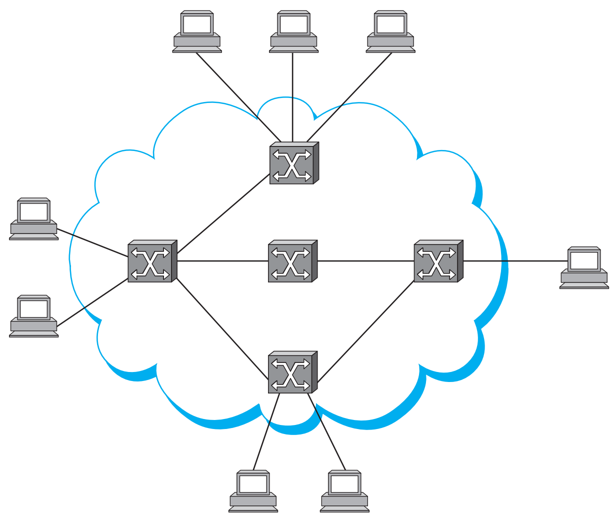
Basic Problem

- Direct-link networks don't scale

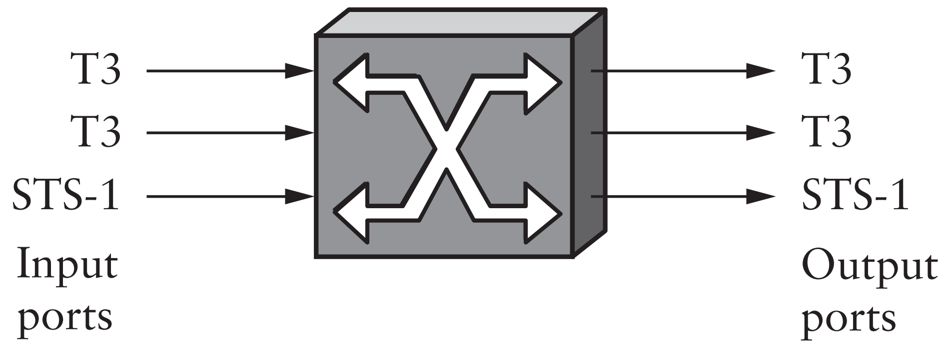




- Solution: use *switches* to connect network segments

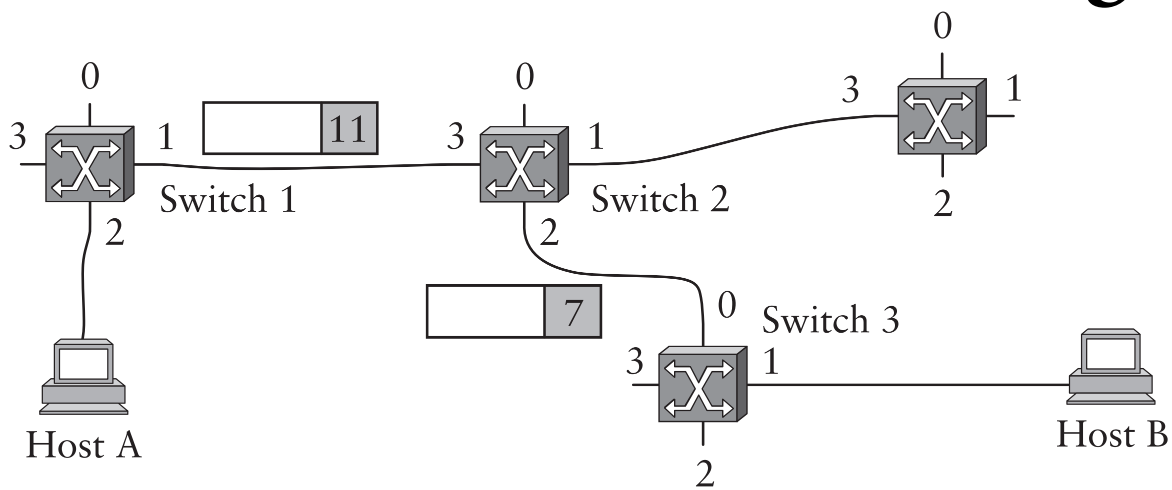


Switching



- **Switches must be able to, given a packet, determine the outgoing port**
- **3 ways to do this:**
  - Datagram Switching
  - Virtual Circuit Switching
  - Source Routing

Virtual Circuit Switching

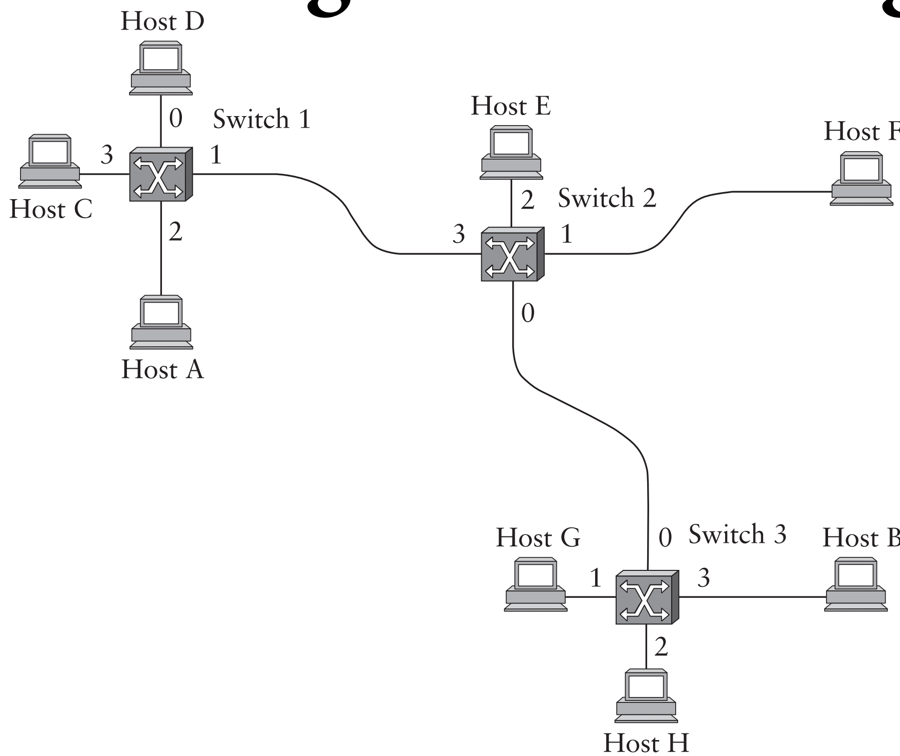


- **Explicit set-up and tear down phases**
  - Establishes Virtual Circuit Identifier on each link
  - Each switch stores VC table
- **Subsequent packets follow same path**
  - Switches map [in-port, in-VCI] : [out-port, out-VCI]
- **Also called *connection-oriented* model**

Virtual Circuit Model

- **Requires one RTT before sending first packet**
- **Connection request contain full destination address, subsequent packets only small VCI**
- **Setup phase allows reservation of resources, such as bandwidth or buffer-space**
  - Any problems here?
- **If a link or switch fails, must re-establish whole circuit**
- **Example: ATM**

Datagram Switching



Switch 2

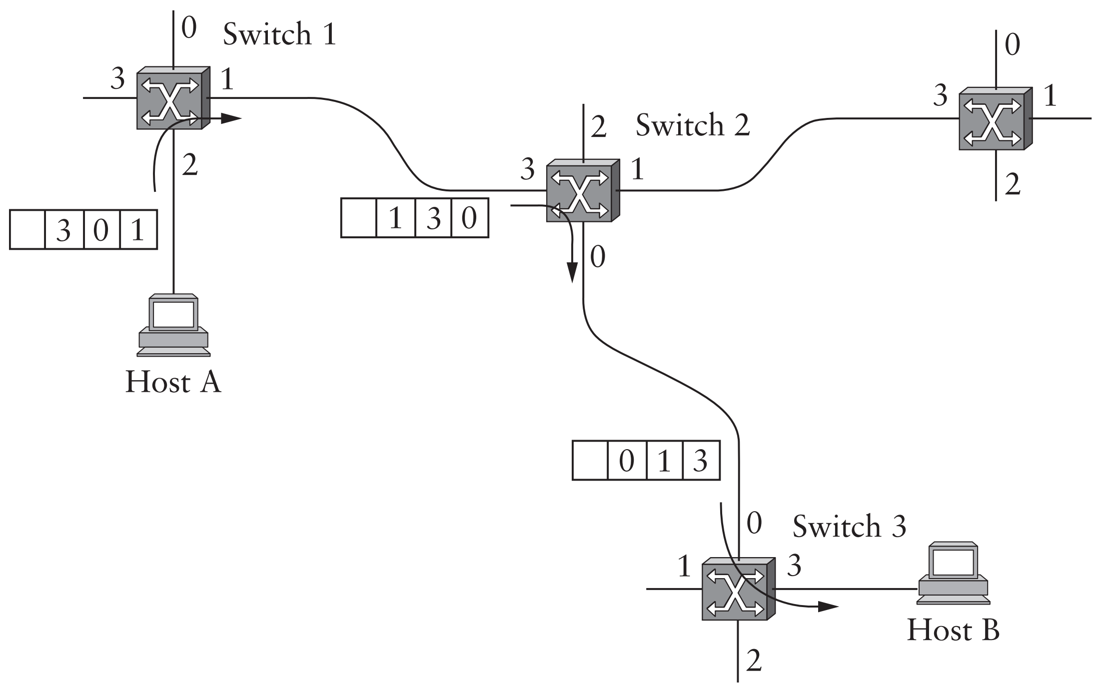
Addr	Port
A	3
B	0
C	3
D	3
E	2
F	1
G	0
H	0

- **Each packet carries destination address**
- **Switches maintain address-based tables**
  - Maps [destination address]:[out-port]
- **Also called *connectionless* model**

Datagram Switching

- No delay for connection setup
- Source can't know if network can deliver a packet
- Possible to route around failures
- Higher overhead per-packet
- Potentially larger tables at switches

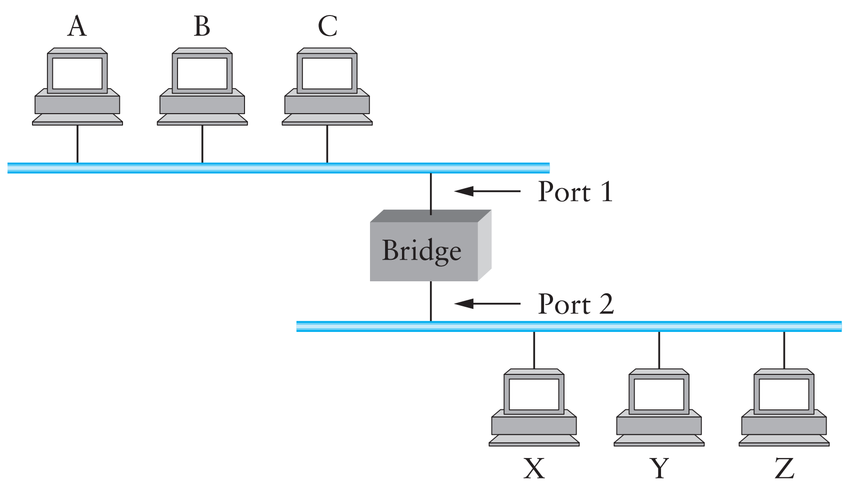
Source Routing



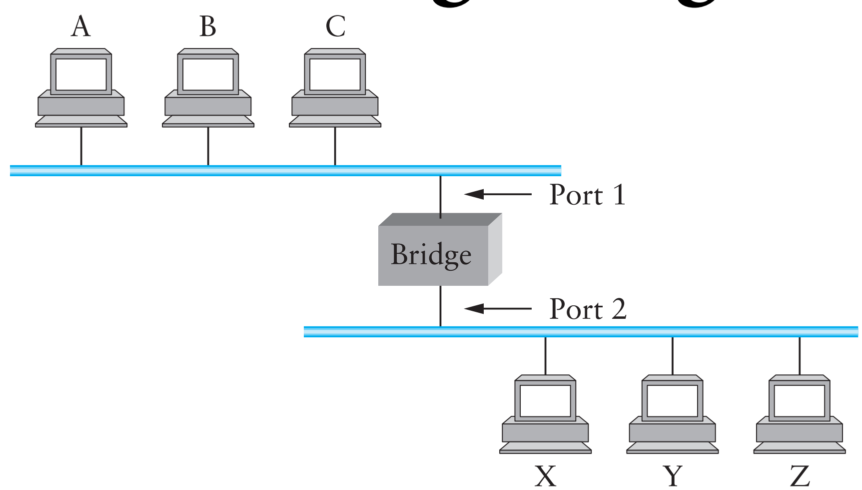
- **Packets carry entire route: ports**
- **Switches need no tables!**
  - But end hosts must obtain the path information
- **Variable packet header**

Bridges and Extended LANs

- **LANs have limitations**
  - E.g. Ethernet < 1024 hosts, < 2500m
- **Connect two or more LANs with a *bridge***
  - Operates on Ethernet addresses
  - Forwards packets from one LAN to the other(s)
- **Ethernet switch is just a multi-way bridge**



Learning Bridges



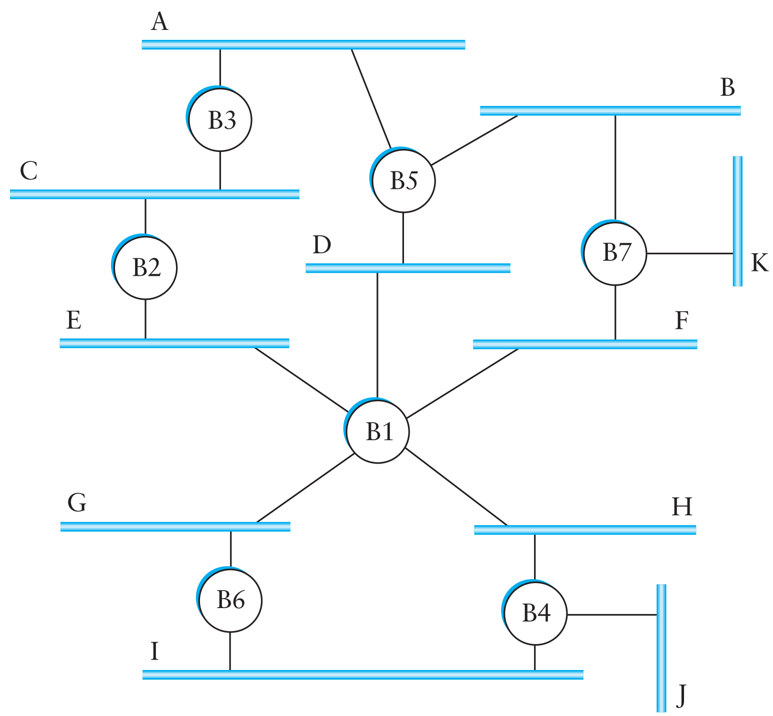
- **Idea: don't forward a packet where it isn't needed**
  - If you know recipient is not on that port
- **Learn hosts' locations based on source addresses**
  - Build a table as you receive packets
- **Table says when *not* to forward a packet**
  - Doesn't need to be complete for *correctness*

Bridges

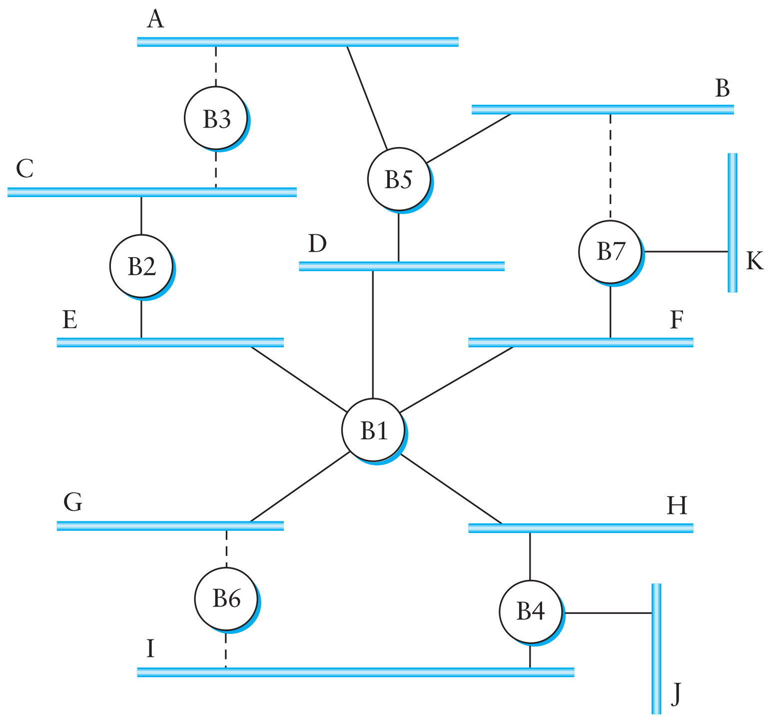
- **Unicast: forward with restrictions**
- **Broadcast: always forward**
- **Multicast: always forward or learn groups**
- **Difference between bridges and repeaters?**
  - Bridges: same broadcast domain; copy *frames*
  - Repeaters: same broadcast and *collision domain*; copy *signals*

Dealing with Loops

- **Problem: people may create loops in LAN!**
  - Accidentally, or to provide redundancy
  - Don't want to forward packets indefinitely



Spanning Tree



- **Need to disable ports, so that no loops in network**
- **Like creating a spanning tree in a graph**
  – View switches and networks as nodes, ports as edges

Distributed Spanning Tree Algorithm

- **Every bridge has a unique ID (Ethernet address)**
- **Goal:**
  - Bridge with the smallest ID is the root
  - Each segment has one designated bridge, responsible for forwarding its packets towards the root
  - Bridge closest to root is designated bridge
  - If there is a tie, bridge with lowest ID wins

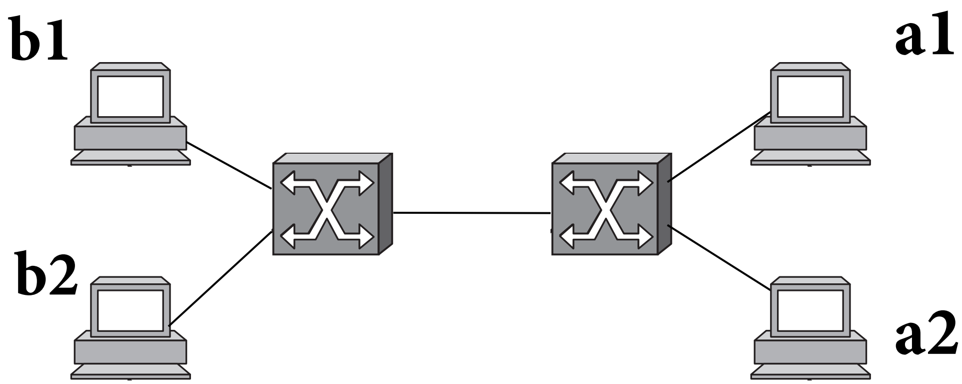
Spanning Tree Protocol

- **Spanning Tree messages contain:**
  - ID of bridge sending the message
  - ID sender believes to be the root
  - Distance (in hops) from sender to root
- **Bridges remember best config msg on each port**
- **Send message when you think you are the root**
- **Otherwise, forward messages from best known root**
  - Add one to distance before forwarding
  - Don't forward if you know you aren't dedicated bridge
- **In the end, only root is generating messages**

Limitations of Bridges

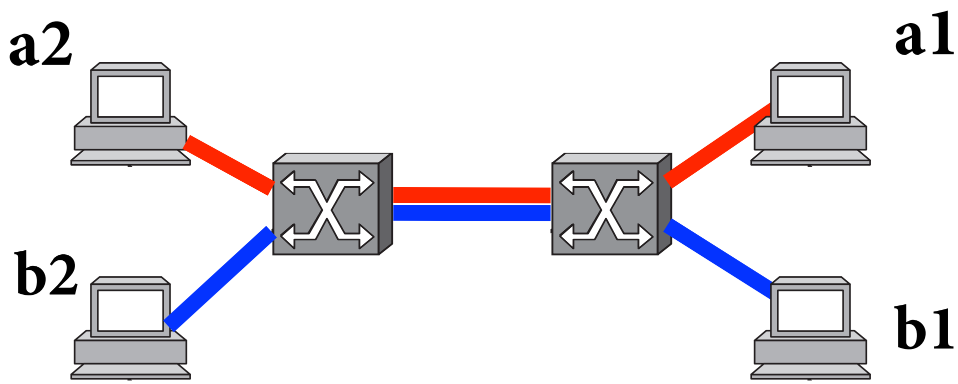
- **Scaling**
  - Spanning tree algorithm doesn't scale
  - Broadcast does not scale
  - No way to route around congested links, *even if path exists*
- **May violate assumptions**
  - Could confuse some applications that assume single segment
  - Much more likely to drop packets
  - Makes latency between nodes non-uniform
  - Beware of transparency

VLANs



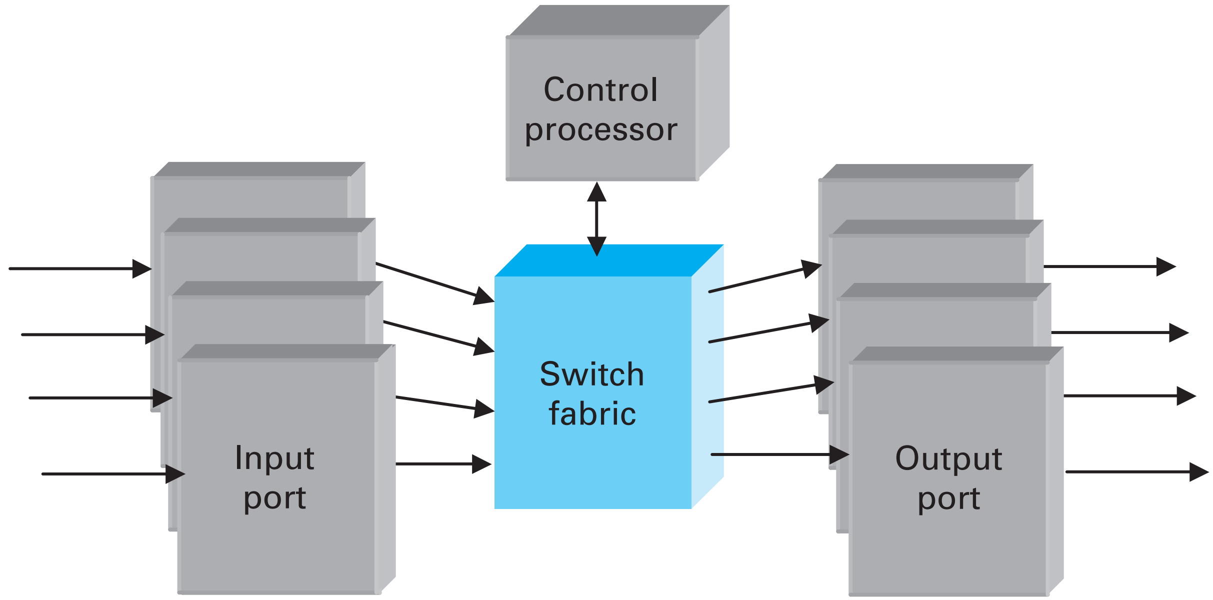
- **Company network, A and B departments**
  - Broadcast traffic does not scale
  - May not *want* traffic between the two departments
  - Topology has to mirror physical locations
  - What if employees move between offices?

VLANs



- **Solution: Virtual LANs**
  - Assign switch ports to a VLAN ID (color)
  - Isolate traffic: only same color
  - Trunk links may belong to multiple VLANs
  - Encapsulate packets: add 12-bit VLAN ID
- **Easy to change, no need to rewire**

Generic Switch Architecture



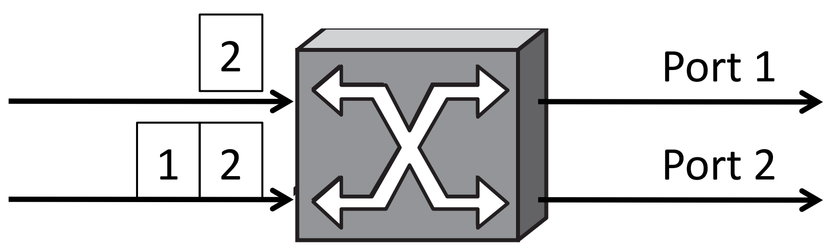
- **Goal: deliver packets from input to output ports**
- **Three potential performance concerns:**
  - Throughput in bytes/second
  - Throughput in packets/second
  - Latency

Cut through vs. Store and Forward

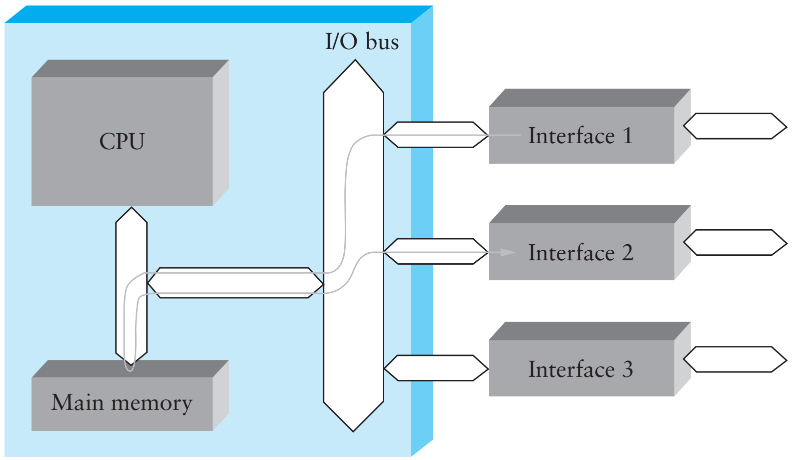
- **Two approaches to forwarding a packet**
  - Receive a full packet, then send to output port
  - Start retransmitting as soon as you know output port, before full packet
- **Cut-through routing can greatly decrease latency**
- **Disadvantage**
  - Can waste transmission (classic *optimistic* approach)
  - CRC may be bad
  - If Ethernet collision, may have to send runt packet on output link

Buffering

- **Buffering of packets can happen at input ports, fabric, and/or output ports**
- **Queuing discipline is very important**
- **Consider FIFO + input port buffering**
  - Only one packet per output port at any time
  - If multiple packets arrive for port 2, they may block packets to other ports that are free
  - *Head-of-line blocking*

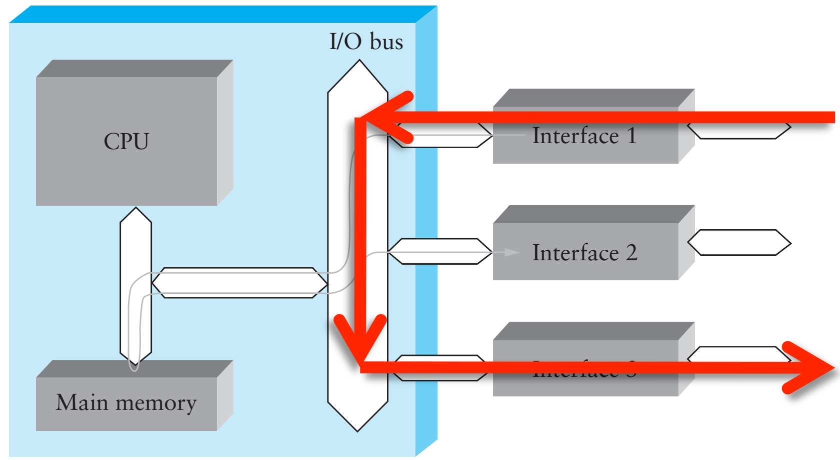


Shared Memory Switch



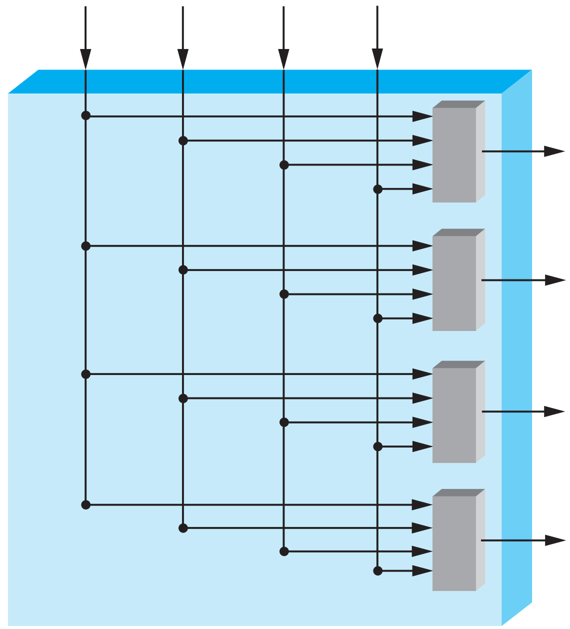
- **1st Generation – like a regular PC**
  - NIC DMAs packet to memory over I/O bus
  - CPU examines header, sends to destination NIC
  - I/O bus is serious bottleneck
  - For small packets, CPU may be limited too
  - Typically < 0.5 Gbps

Shared Bus Switch



- 2^{st} **Generation**
  - NIC has own processor, cache of forwarding table
  - Shared bus, doesn't have to go to main memory
  - Typically limited to bus bandwidth
    - (Cisco 5600 has a 32Gbps bus)

Point to Point Switch



- **3rd Generation: overcomes single-bus bottleneck**
- **Example: Cross-bar switch**
  - Any input-output permutation
  - Multiple inputs to same output requires trickery
  - Cisco 12000 series: 60Gbps

Coming Up

- Let's connect multiple networks: IP and the Network Layer
- Remember: no class on Tuesday!