

Router Construction

Outline

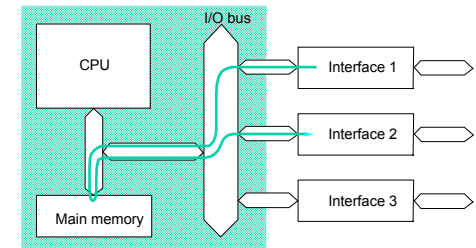
Switched Fabrics

IP Routers

Tag Switching

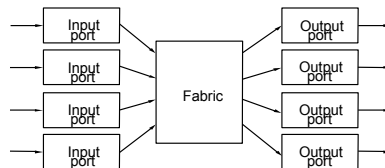
Workstation-Based

- Aggregate bandwidth
 - 1/2 of the I/O bus bandwidth
 - capacity shared among all hosts connected to switch
 - example: 1Gbps bus can support 5 x 100Mbps ports (in theory)
- Packets-per-second
 - must be able to switch small packets
 - 1M packets-per-second is achievable
 - e.g., 64-byte packets implies 622Mbps



Switching Hardware

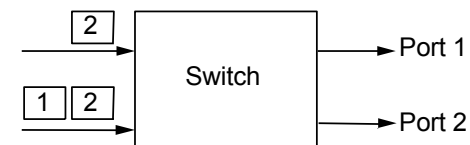
- Design Goals
 - throughput (depends on traffic model)
 - scalability (a function of n)



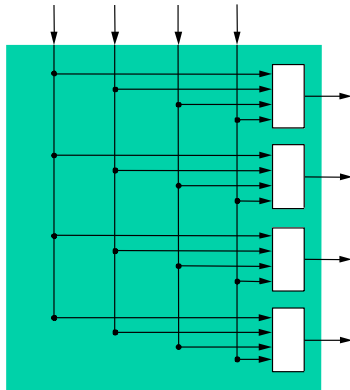
- Ports
 - circuit management (e.g., map VCIs, route datagrams)
 - buffering (input and/or output)
- Fabric
 - as simple as possible
 - sometimes do buffering (internal)

Buffering

- Wherever contention is possible
 - input port (contend for fabric)
 - internal (contend for output port)
 - output port (contend for link)
- Head-of-Line Blocking
 - input buffering



Crossbar Switches



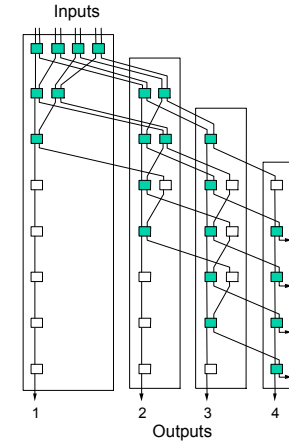
Spring 2005

CS 461

5

Knockout Switch

- Example crossbar
- Concentrator
 - select l of n packets
- Complexity: n^2



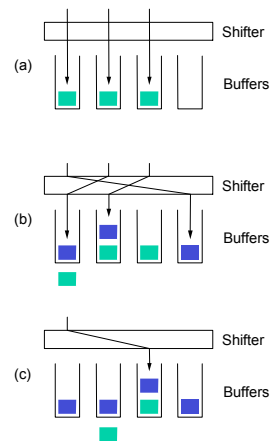
Spring 2005

CS 461

6

Knockout Switch (cont)

- Output Buffer



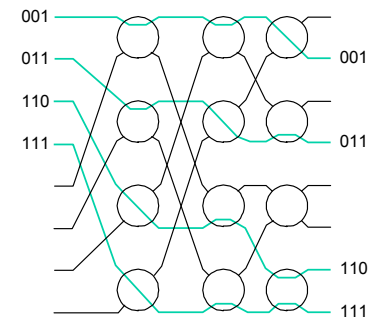
Spring 2005

CS 461

7

Self-Routing Fabrics

- Banyan Network
 - constructed from simple 2×2 switching elements
 - self-routing header attached to each packet
 - elements arranged to route based on this header
 - no collisions if input packets sorted into ascending order
 - complexity: $n \log_2 n$



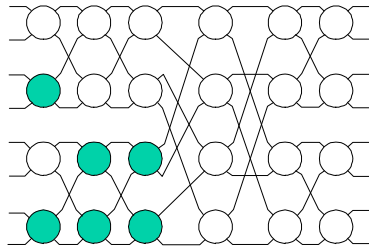
Spring 2005

CS 461

8

Self-Routing Fabrics (cont)

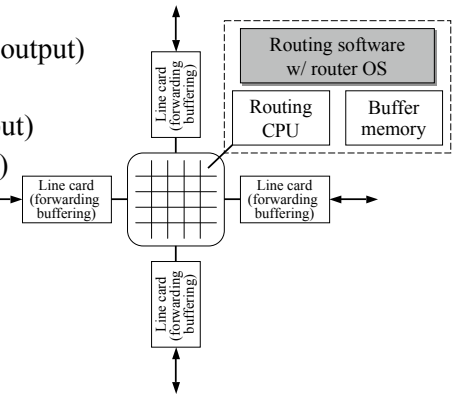
- Batcher Network
 - switching elements sort two numbers
 - some elements sort into ascending (clear)
 - some elements sort into descending (shaded)
 - elements arranged to implement merge sort
 - complexity: $n \log^2_2 n$



- Common Design: Batcher-Banyan Switch

High-Speed IP Router

- Switch (possibly ATM)
 - link interface (input, output)
 - router lookup (input)
 - common IP path (input)
 - packet queue (output)
- Line Cards
 - link interface (input, output)
 - router lookup (input)
 - common IP path (input)
 - packet queue (output)
- Control Processor
 - routing protocol(s)
 - exceptional cases



IP Forwarding is Slow

- Problem: classless IP addresses (CIDR)
- Route by variable-length Forwarding Equivalence Classes (FEC)
 - FEC = IP address plus prefix of 1-32 bits; e.g., 172.200.0.0/16
- IP Router
 - forwarding tbl: $\langle \text{FEC} \rangle \rightarrow \langle \text{next hop, port} \rangle$
 - match IP address to FEC w/ longest prefix

ATM Forwarding

- Primary goal: fast, cheap forwarding
- 1Gb/s IP router: \$187,000
- 5Gb/s ATM switch: \$41,000
- Create Virtual Circuit at Flow Setup
 - $\langle \text{in VCI} \rangle \rightarrow \langle \text{port, out VCI} \rangle$
- Cell Forwarding
 - index, swap, switch

Tag Switching (MPLS)

- Add a VCI-like tag to packets
 - <in tag> → <next hop, port, out tag>
- Use ATM switch hardware
- IP routing protocols (OSPF, RIP, BGP)
 - build forwarding table from routing table
- Goal: IP router functionality at ATM switch speeds/costs

Forwarding

- *Shim* before IP header



- Tag Forwarding Information Base (TFIB)
 - <in tag> → <next hop, port, out tag>
- Just like ATM
 - index, swap, switch

Tag Binding

- New FEC from IP routing protocols
 - Select local tag (index in TFIB)
 - <in tag> → <next hop, port, ???>
- Need <out tag> for next hop
- Other routers need my <in tag>
- Solution: distribute tags like other routing info

Tag Distribution Protocol

- Send TDP messages to peers
 - <FEC, my tag>
- Upon receiving TDP message, check if sender is next hop for FEC
 - yes, save tag in TFIB
 - no, can discard or save for future use
- ‘Control-driven’ label assignment

The First Tag

- Two kinds of routers: edge vs. interior



- Edge: add shim based on IP lookup, strip at exit
- Interior: forward by tag only

Robustness Issues

- What if tag fault?
 - try to forward (default route)
 - discard packet
- Forwarding Loops
 - topology changes cause temporary loops
 - TTL field in tag, same as IP

Ipsilon: IP Switching

- Run on ATM switch over ATM network
 - ATM hardware + IP switching software
- Idea: Exploit temporal locality of traffic to cache routing decisions
- Associate labels (VCI) with flows
 - forward packets as usual
 - main difference is in how labels are created, distributed to other routers

Alternative: IP Switch

- Assume default ATM virtual circuits between routers
- Router runs IP routing protocol, can forward IP packets on default VCs
- Identify flows, assign flow-specific VC
 - flow = port pair or host pair
- ‘Data-driven’ label assignment