Interconnection networks

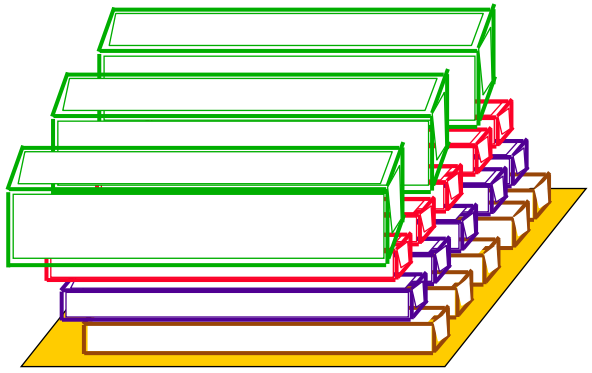
6.173
Fall 2010
Agarwal

Outline

- Topology and switch architecture
  - Arrangement of nodes, edges (switches, wires)
- Flow control
  - Allocation of node, channel resources
- Routing
  - Choosing paths
- Addressing and switch design
  - Mapping of names to physical locations
- Performance
  - Latency
  - Bandwidth
  - Efficient 2D layout

Remember the wire throughout this discussion!
We will start with a VLSI wire primer

VLSI Wire Primer

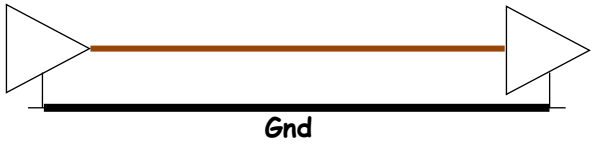


Chip surface

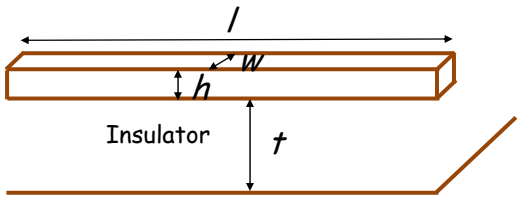
Limit on number of wires that can be supported

E.g., 10 layers

VLSI Wire Primer



Metal line (Aluminium, copper, tungsten)



Energy to switch state relates to $\frac{1}{2}CV^2$

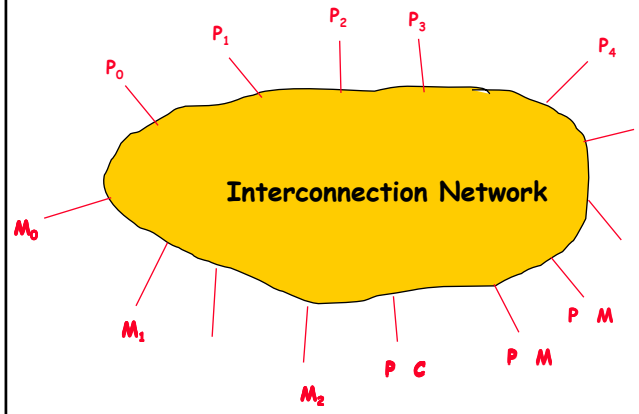
$$C = \frac{\varepsilon_{ox} lw}{t}$$ (e.g, 100fJ per mm)

Delay relates to RC

$$RC = \frac{\rho l}{wh} \frac{\varepsilon_{ox} lw}{t}$$

They also serve who stand and communicate!
Sorry Milton

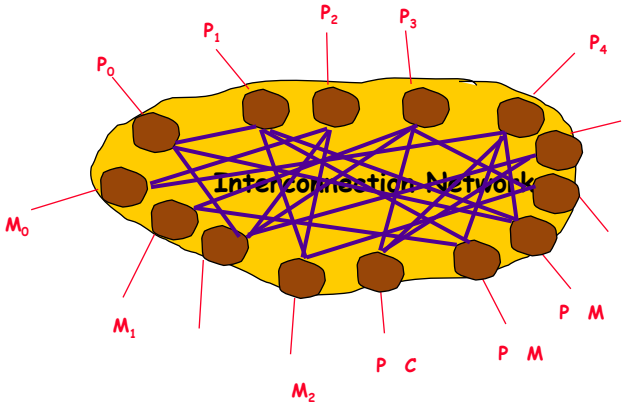
The Big Picture



Many choices
Processors connected to memory over network
Processor-cache cores connected to each other
processor-memory nodes connected to each other

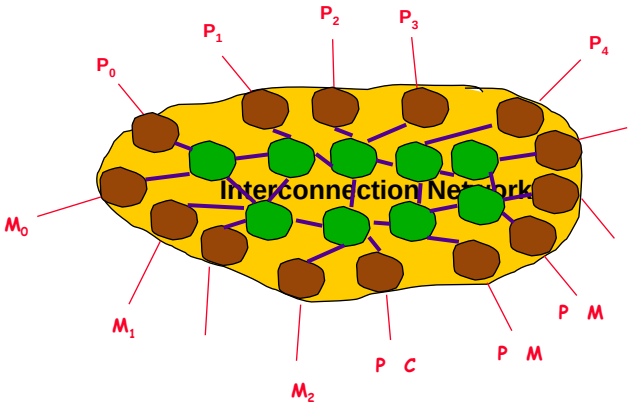
Topology

Topology



Direct network
All network nodes have processor (or memory) attached
In other words, direct connection between procs

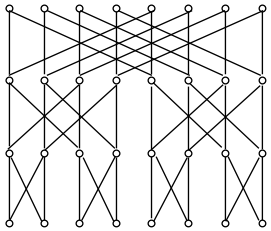
Topology



Indirect network
Intermediate routing-only nodes
No direct connection between processors

Topology

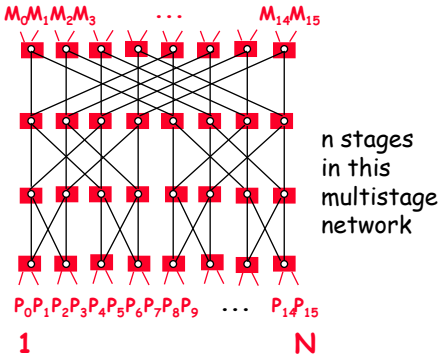
More Direct vs Indirect networks



Topology

Direct vs Indirect networks

Indirect network

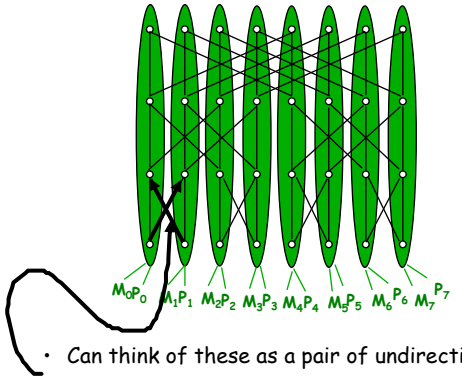


- Has routing-only nodes
- Lots of hardware
- Few pins per switch node
- Average hops $Log_2 N = n$
- *Latency, bandwidth?*

Topology

Direct vs Indirect networks

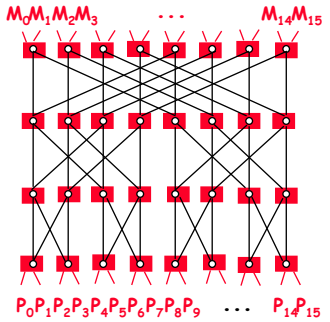
Direct Network



- Can think of these as a pair of undirectional links
- Direct, processor-to-processor connections
- Fewer wires
- More wires per node
- Can think of indirect network as a pipelining of each direct network node
- Latency, bandwidth?

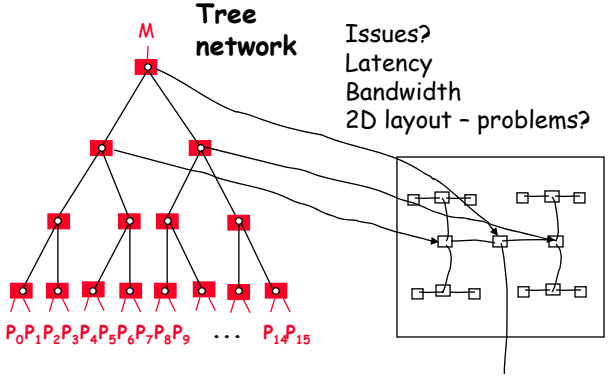
Topology

Indirect network examples



Butterfly network

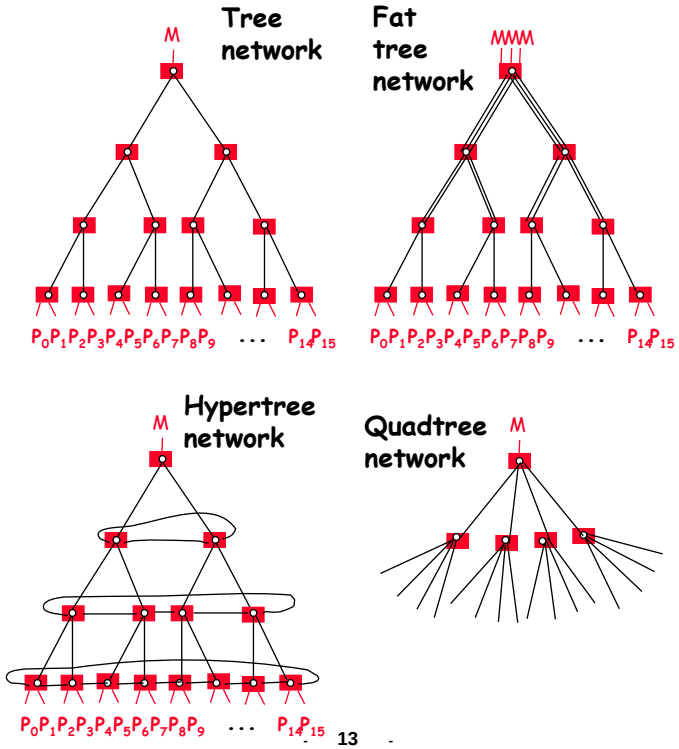
Tree network



Issues?
Latency
Bandwidth
2D layout – problems?

Topology

Indirect network examples



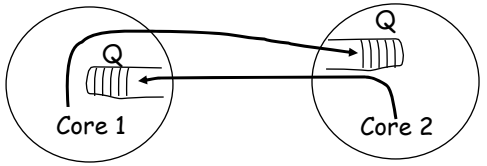
Topology -- Direct Networks

k-ary n-cube networks

radix dimension

k=2

n=1
1 cube



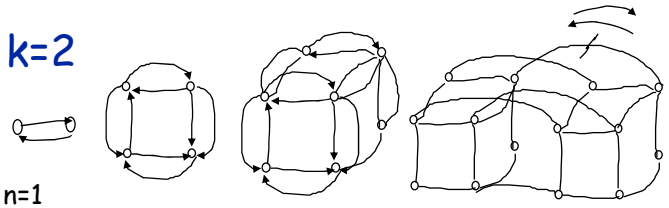
Two cores
With channels in both directions

Topology -- Direct Networks

k-ary n-cube networks

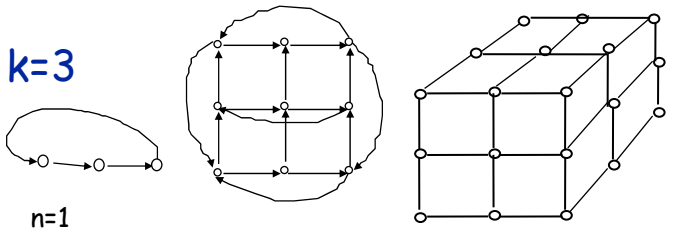
radix dimension

k=2



n=1 1 cube | 2 cube | 3 cube | 4 cube ...

k=3



n=1 1 cube | 2 cube | 3 cube

aka ring | aka mesh | aka ring

Nodes $N = k^n$

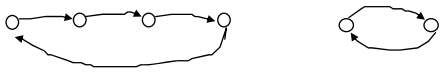
Hypercube = binary *n*-cube *(k=2)*

Topology -- Direct Networks

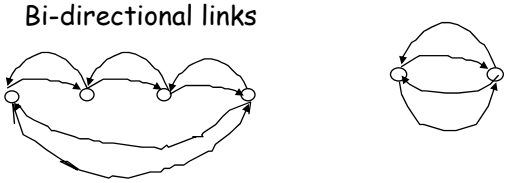
k-ary n-cube networks

Variants:

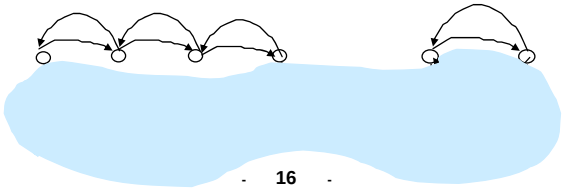
Uni-directional links



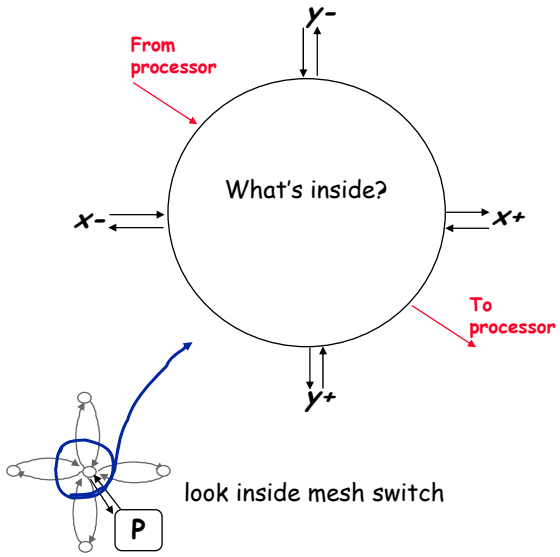
Bi-directional links



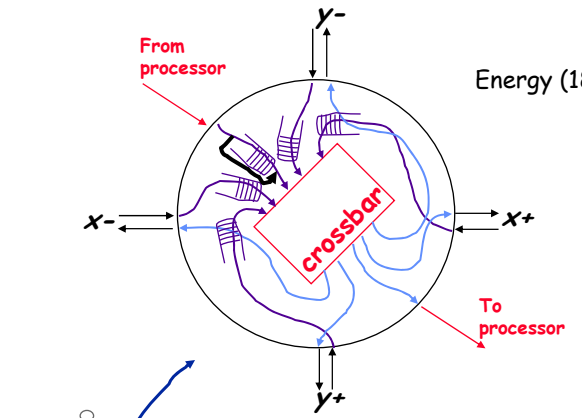
Bi-directional links; no end-around

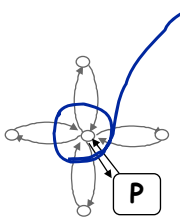


Switch Architecture



Switch Architecture
What's Inside a Switch

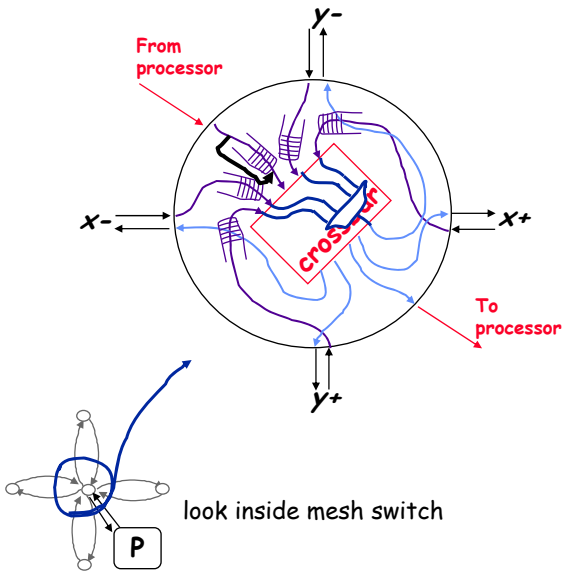




look inside mesh switch

Buffers can be added to input posts, or output ports, or both

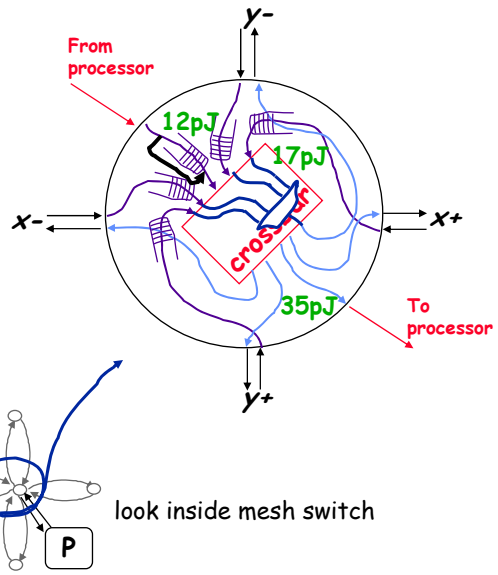
Switch Architecture
What's inside the crossbar



look inside mesh switch

Crossbar can be built using a 4-input mux for each output

Switch Architecture
Energy



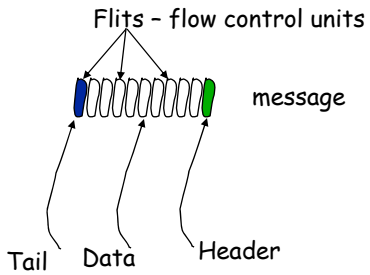
look inside mesh switch

Crossbar can be built using a 4-input mux for each output

(Raw chip 180nm)

Flow Control (or Switching)

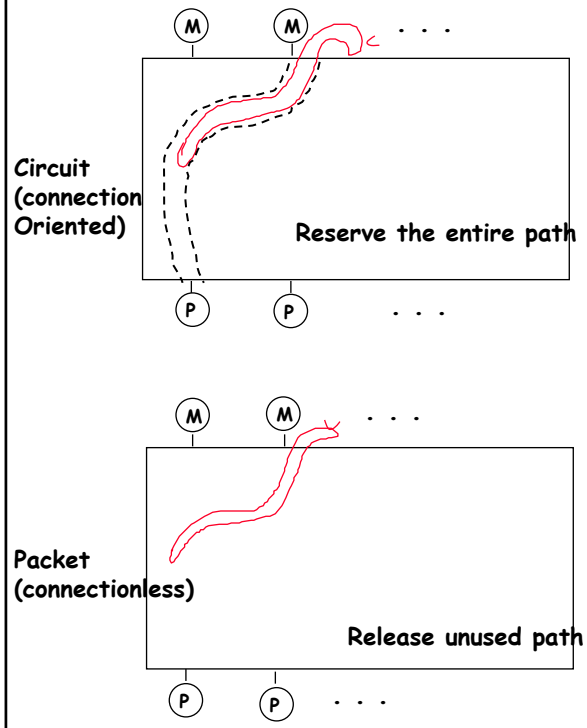
Notation



We will denote a message using this cartoon

Flow Control

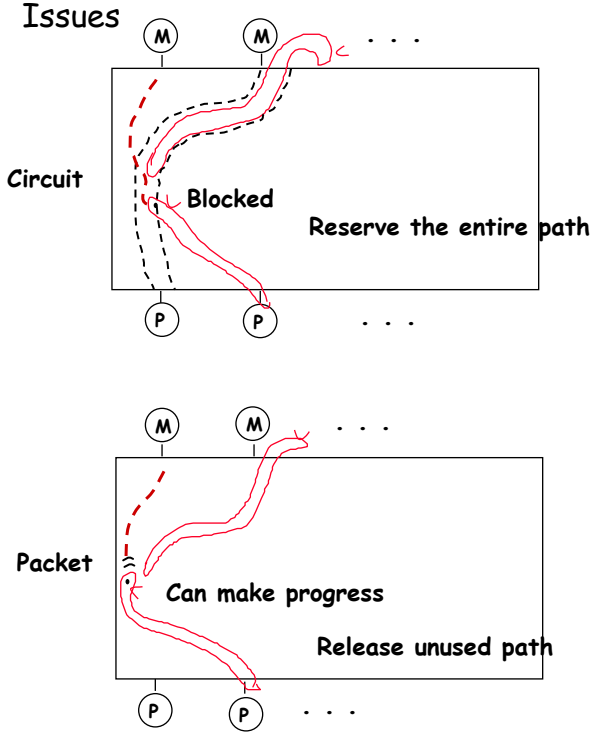
Circuit-switched vs. Packet-switched



Flow Control

Circuit-switched vs. Packet-switched

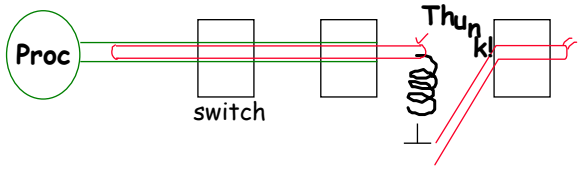
Issues



Flow Control

What do you do when you block?

- Dropping or non blocking
  - Commonly used with circuit switching



  - Reserve resources along entire path from source to destination
  - Ack has reserved return path!
  - Drop when blocked (non-blk network)
  - Tail can reverse path for NACK along the reserved backpath
  - Retry - can use backoffs
  - No need for buffering, packet is always moving forward (even if it is to nowhere!)

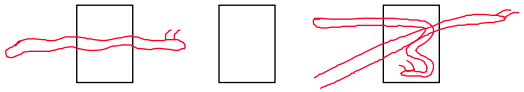
Flow Control

Mostly packet switched - release resources

- Store & forward

  - Always store msg. completely at each node, even if outport is free. Then send

- Cut through



  - Do not store message if outport is free; buffer only if outport is busy

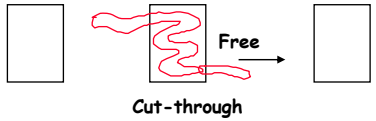
Two variants of cut through

- **1. virtual cut through: flow control on packet**
- **2. pipelined or wormhole: flow control on flits**

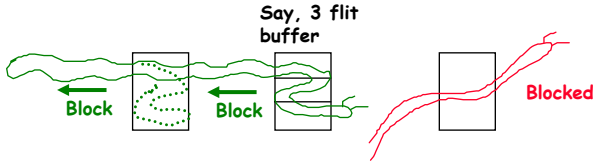
Flow Control

Cut through approaches

- 1. Virtual cut-through (no backward flow control)

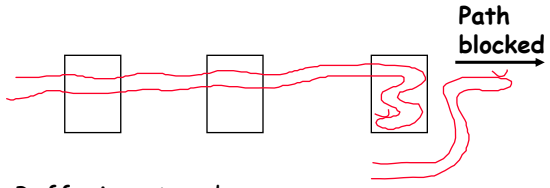


  - Send out as soon as outport is free
  - Need enough buffer space for whole msg -- no blocking
- 2. Cut-through with backward flow control (piplelined, wormhole) works with finite buffers

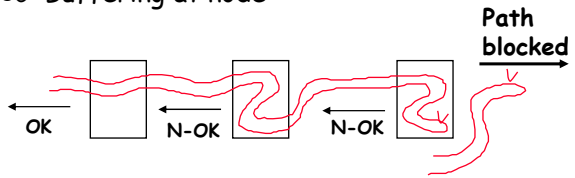


Flow Control

Cut through summary: two ways of message blocking



∞ Buffering at node



Partial buffering in node, partial over network

← Need backward flow-control signals