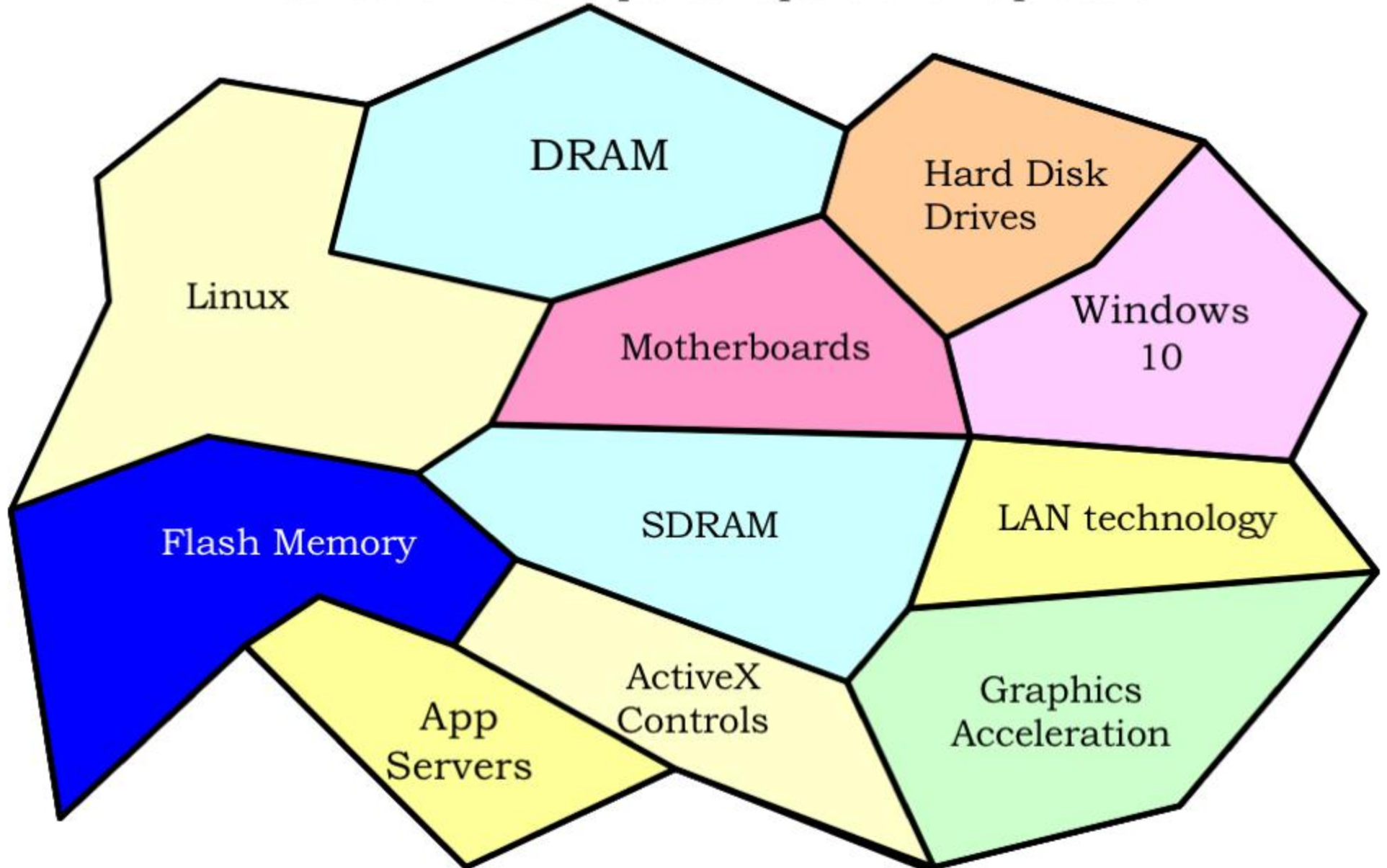


Computer System Technologies

What is the most important part of this picture?

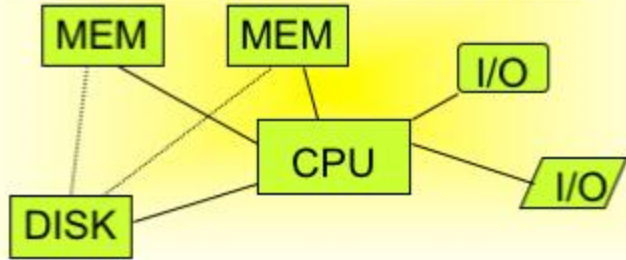


Technology comes & goes; interfaces last forever

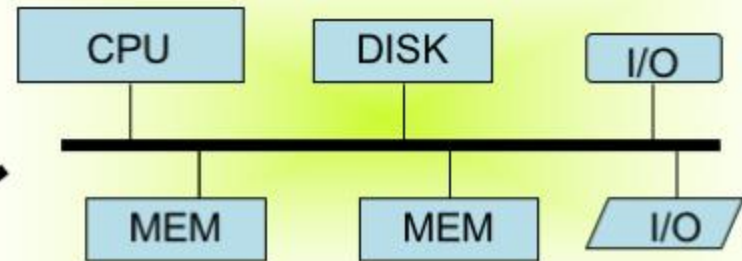
- Interfaces typically deserve more engineering attention than the technologies they interface...
 - Abstraction: should outlast many technology generations
 - Often “virtualized” to extend beyond original function (e.g. memory, I/O, services, machines)
 - Represent more potential value to their proprietors than the technologies they connect.
- Interface sob stories:
 - Interface “warts”: Big/little Endian wars
 - Early IBM PC reliance on the exact signaling of 8086 chips
- ... and many success stories:
 - IBM 360 Instruction set architecture; Postscript; Compact Flash; ...
 - TCP/IP-based packet networks

System Interfaces & Modularity

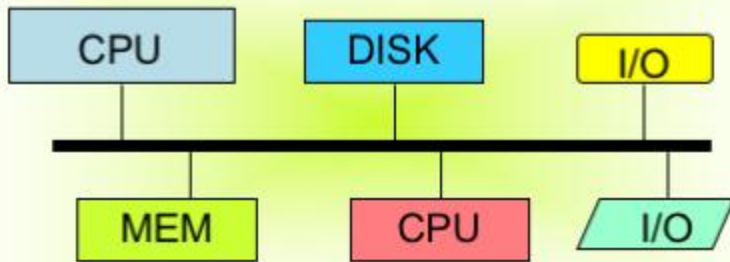
Ancient Times (Ad hoc connections)



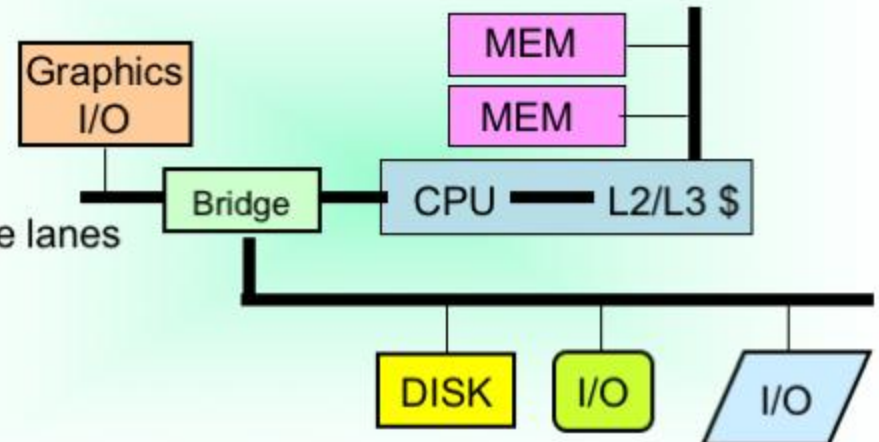
Late 60s (Processor-dependent Bus)



80s (Processor-independent Bus)



90s (buses galore)



(Mostly) Point-to-point

Buses, Interconnect, So...?

Aren't communication channels simply logic circuits with long wires?

Wires – circuit theorist's view:

Equipotential “nodes” of a circuit.

Instant propagation of v , i over entire node.

“distance” abstracted out of design model.

Time issues dictated by RLC elements; wires are timeless.



Wires – interconnect engineer's view:

Transmission lines.

Finite signal propagation velocity.

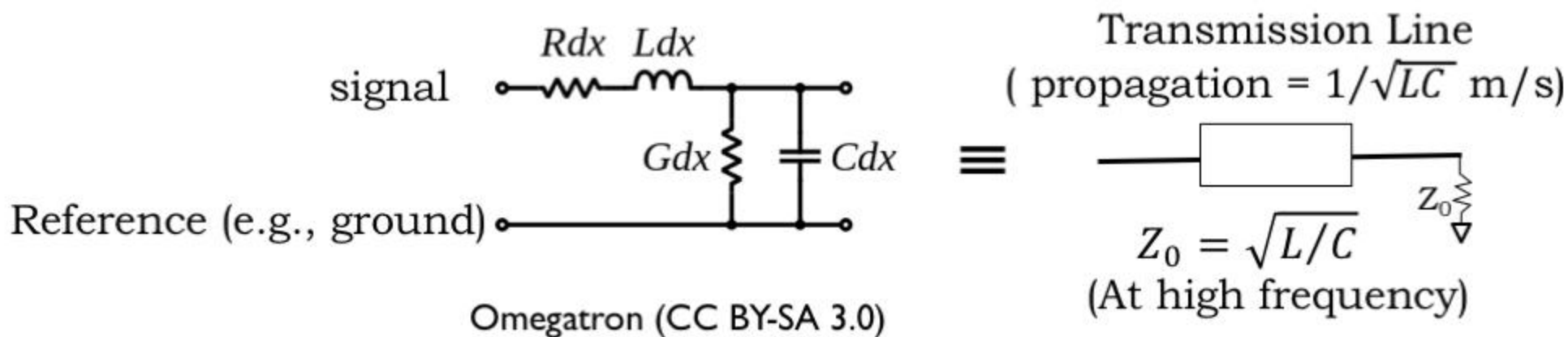
Distance matters.

Time matters.

Reality matters.



Electrical Model for Real Wires

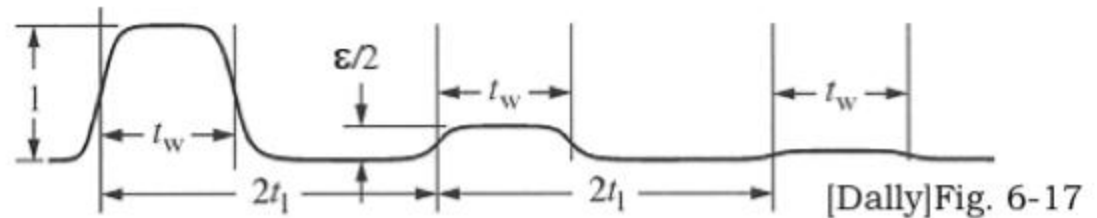
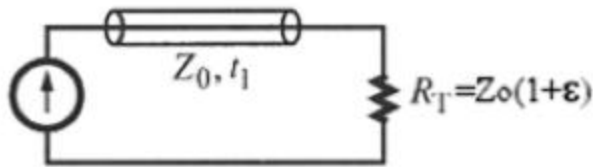


	Description	On chip	On PCB
R	Resistance of conductor	150kΩ/m	5Ω/m
L	Self-inductance of conductor (due to magnetic field induced by current)	600nH/m	300nH/m
C	Capacitance between signal and ground	200pF/m	100pF/m
G	Conductance between signal and ground (through insulator)	small	small

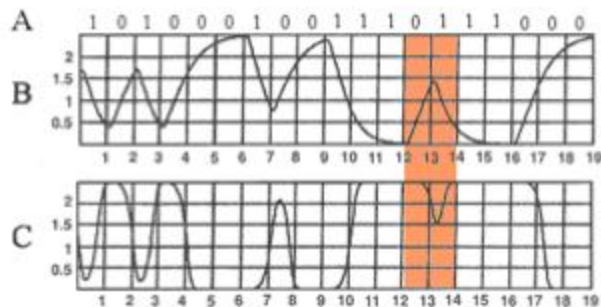
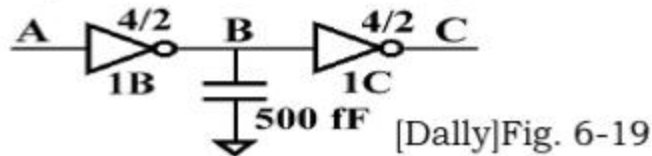
Real-World Consequences

ΔV from energy storage left over from earlier signaling on the wire:

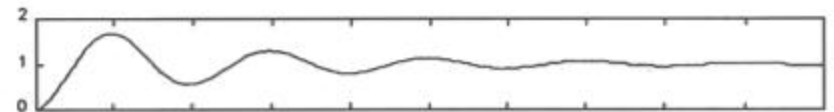
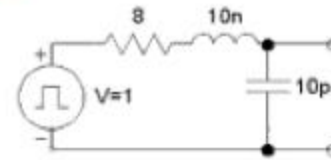
- transmission line discontinuities**
(reflections off of impedance mismatches and terminations)



- charge storage in RC circuit** (narrow pulses are lost due to incomplete transitions)
- RLC ringing** (triggered by voltage steps)



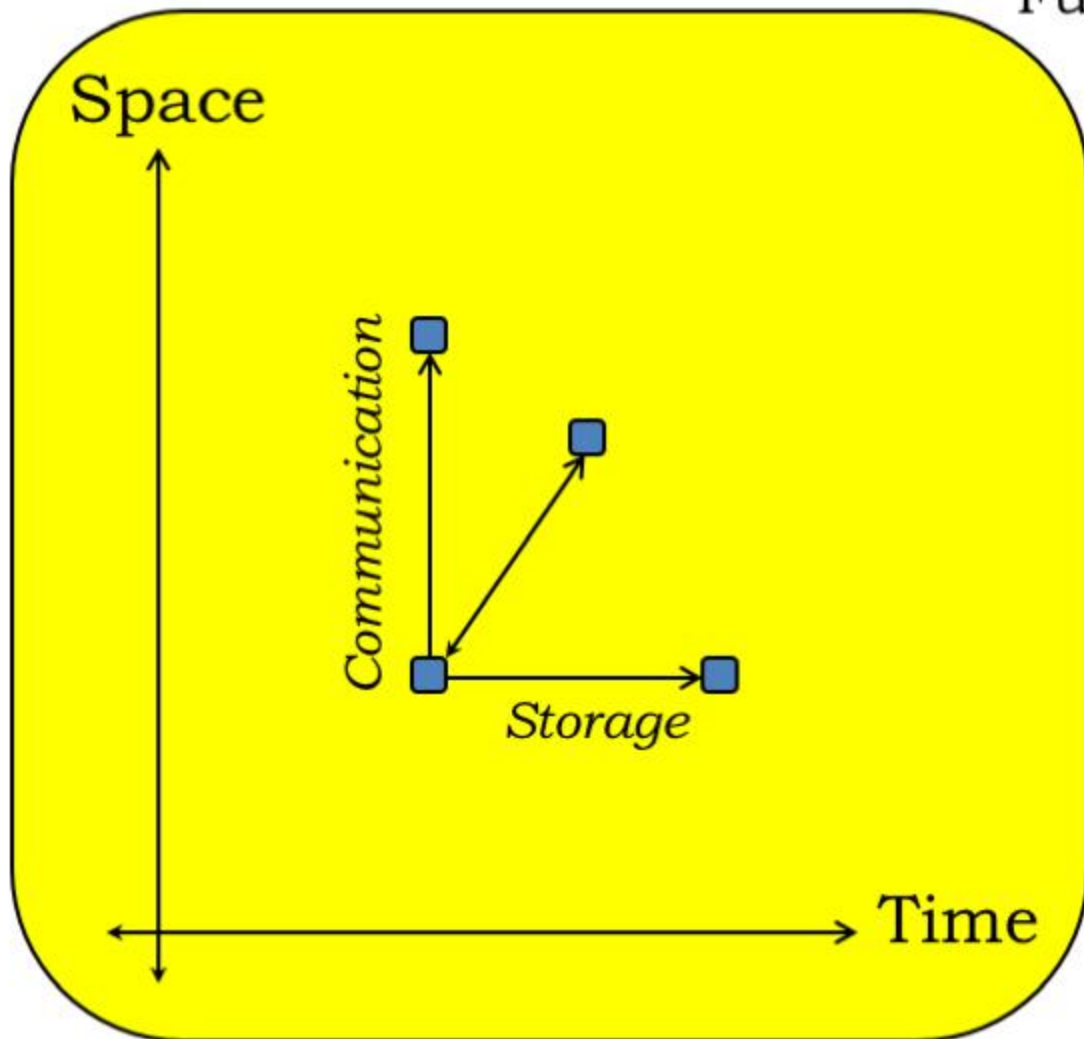
[Dally]Fig. 6-20



Fix: slower operation, limiting voltage swings and slew rates

Space & Time Constraints

Fundamental Physical Constraints:

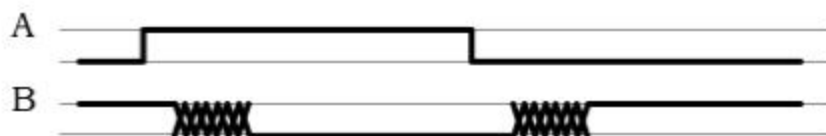


- Bounds on propagation speeds
 - Signals travel $\sim 18\text{cm/ns}$ on PCB
- Bounds on device density
 - Must be finite distances between components
- Bounds on flow of charge
 - finite currents $\rightarrow$ finite rise/fall times
 - wire delays depend on loading

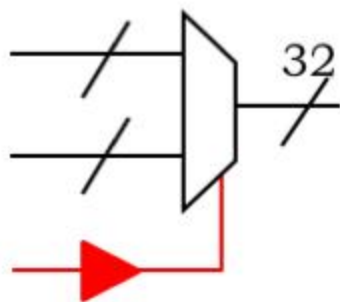
Gates, Wires, & Delays

Our t_{pd} , t_{cd} timing model

- bundles delays into device specs
- ignores loading, wire lengths



Reality check:



- long / heavily-loaded outputs will be slower
- can bundle internal wire delays into t_{pd} of a device; but external load matters!
- partial fixes: buffers, distribution trees
- optimizing performance requires attention to loading issues (You'll see this in the design project!).

Particularly problematic: system-wide interconnect!

Interface Standard: Backplane Bus

Modular cards that plug into a common backplane:

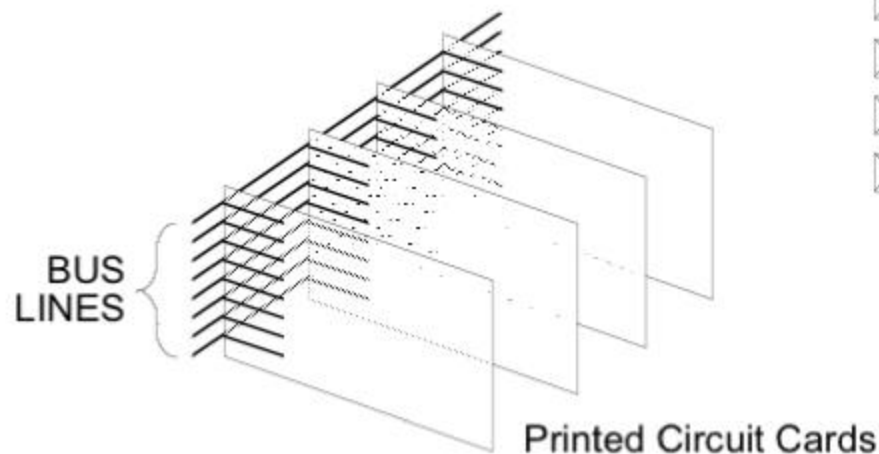
CPU's

Memories

Bulk storage

I/O devices

S/W?

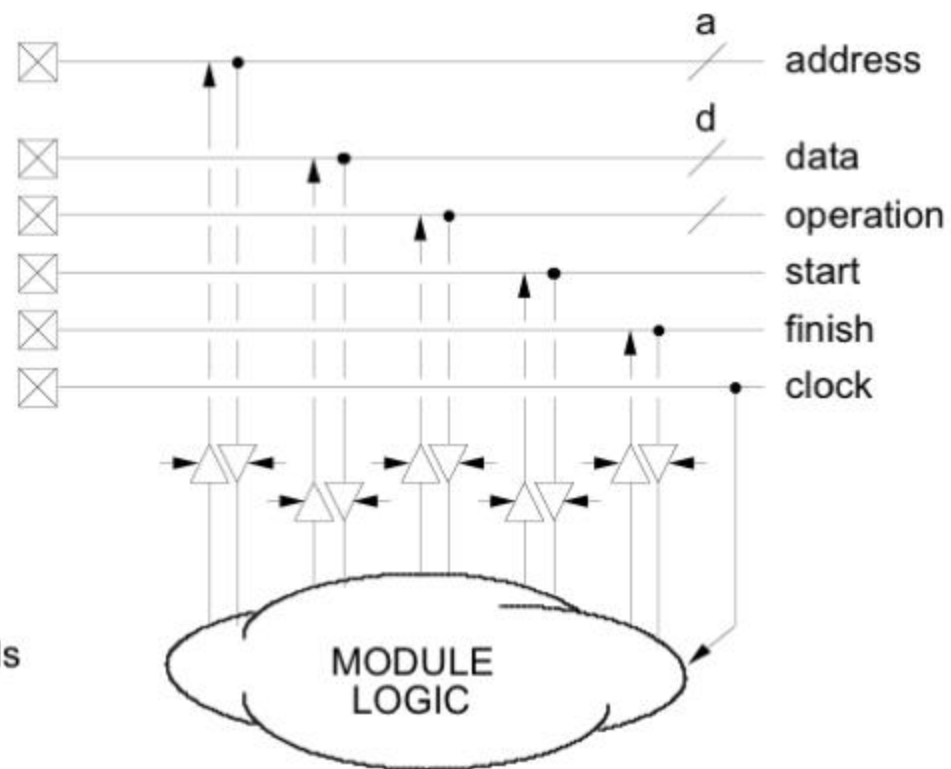


The backplane provides:

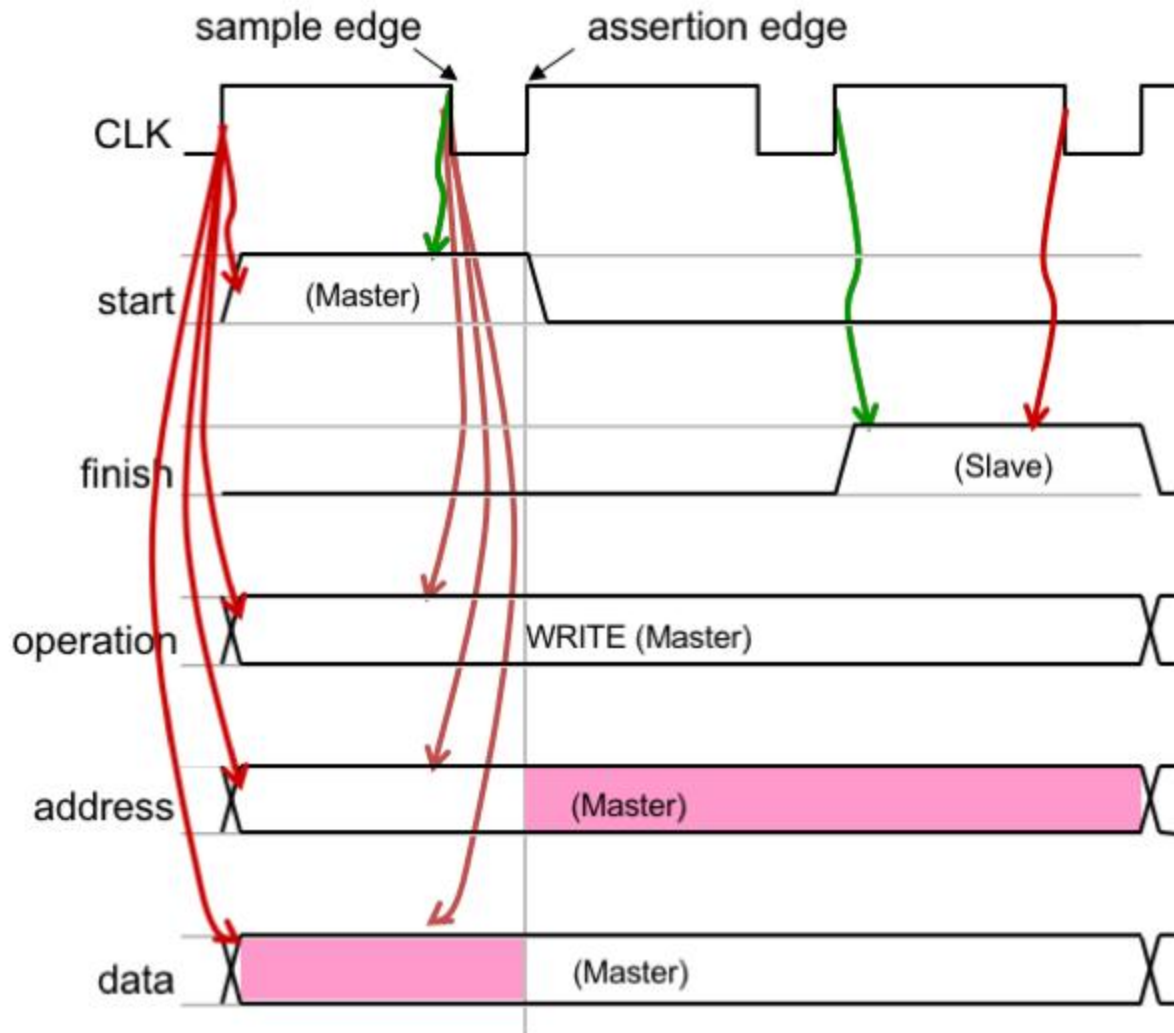
Power

Common system clock

Wires for communication



A Parallel Bus Transaction



MASTER:

- 1) Chooses bus operation
- 2) Asserts an address
- 3) Waits for a slave to answer.

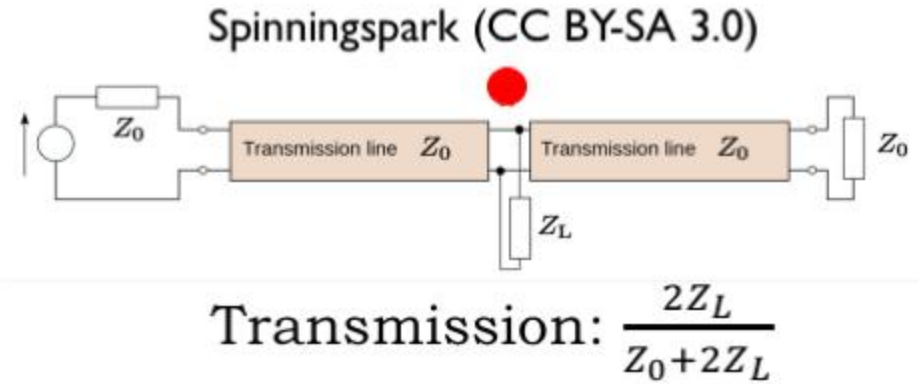
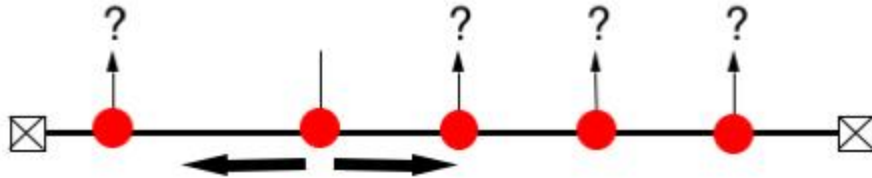
SLAVE:

- 1) Monitors start
- 2) Check address
- 3) If meant for me
 - a) look at bus operation
 - b) do operation
 - c) signal finish of cycle

BUS:

- 1) Monitors start
- 2) Start count down
- 3) If no one answers before counter reaches 0 then "time out"

Bus Lines as Transmission Lines



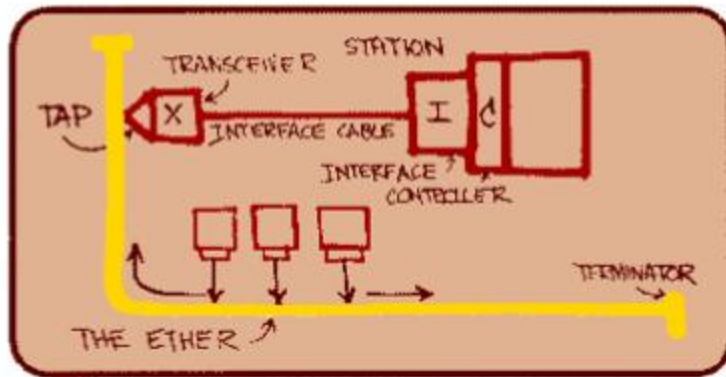
ANALOG ISSUES:

- Propagation times
 - Signals travel at ~ 18 cm/ns on a PCB
- Skew
 - Different points along the bus see the signals at different times
 - Bits of data propagate at slightly different rates along parallel wires
- Reflections & standing waves
 - At each interface (places where the propagation medium changes) the signal may reflect if the impedances are not matched.
 - Make a transition on a long line – may have to wait many transition times for echoes to subside.

Meanwhile, Outside the Box...

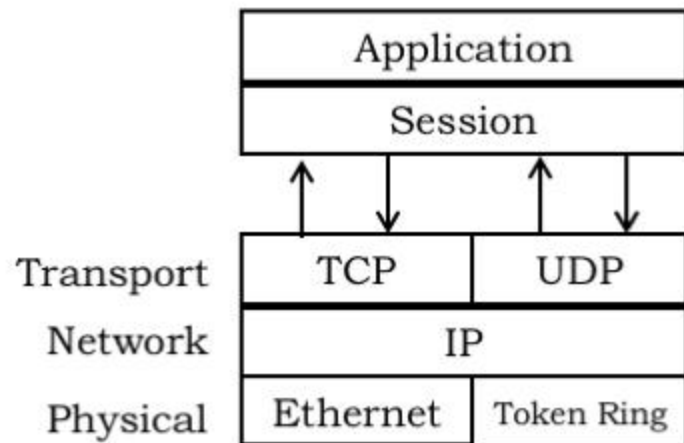
The network as an interface standard.

ETHERNET: In the mid-70's Bob Metcalf (at Xerox PARC, an MIT alum) devised a bus for networking computers together.



- Inspired by Aloha net (radio)
- COAX replaced “ether”
- *Bit-serial* (optimized for long wires)
- Variable-length “packets”:
 - self-clocked data (no clock, skew!)
 - header (dest), data bits, checksum
- Issues: sharing, contention, arbitration, “backoff”

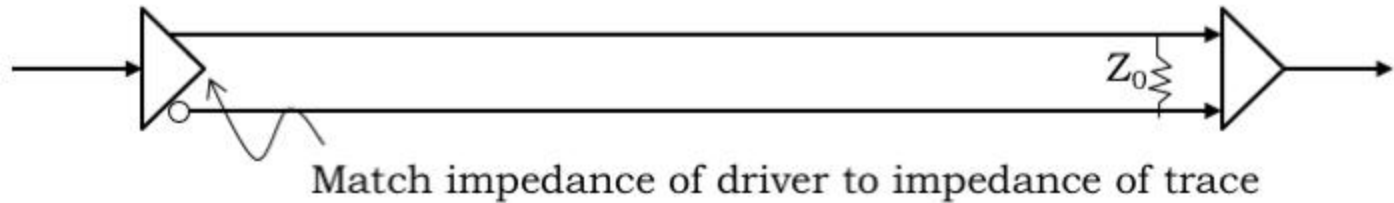
IDEA: Protocol “layers” that isolate application-level interface from low-level physical devices:



Lessons learned: single driver, point-to-point

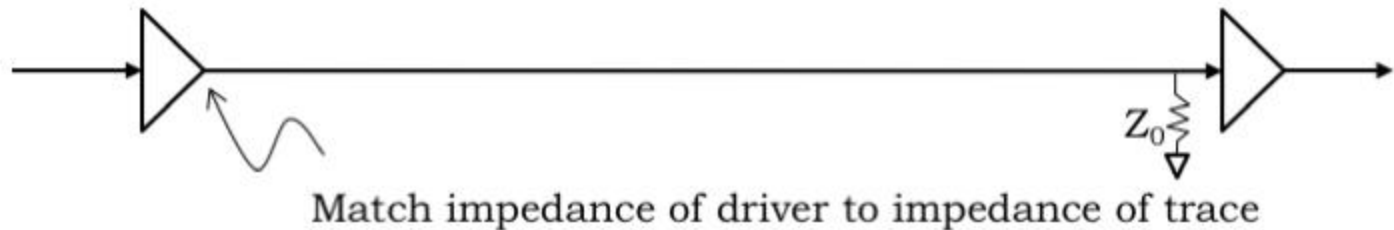
Differential signaling over controlled impedance trace

BEST

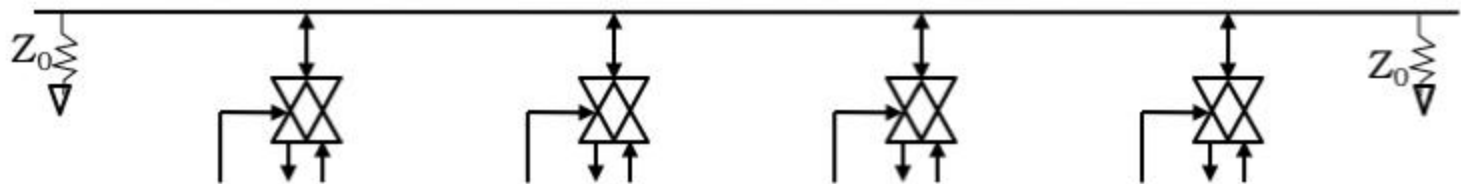


Single-ended signaling over controlled impedance trace

OKAY



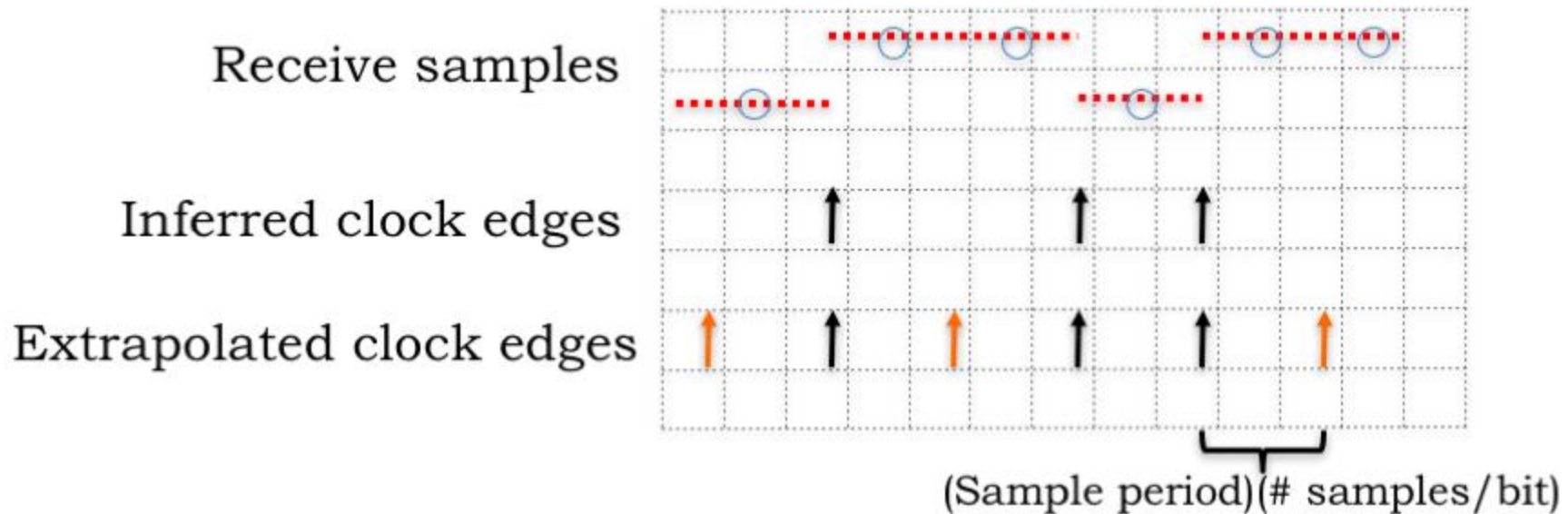
SLOW



Issues:

- Impedance troubles when driving in middle
- Turn-around time when sharing a wire (wired-or glitch)

Lessons learned: clock recovery

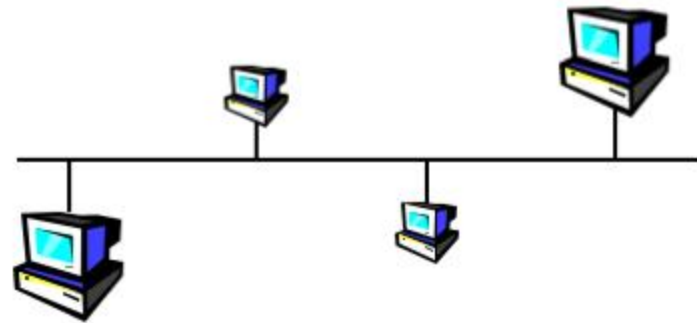


- Receiver can infer presence of clock edge every time there's a transition in the received samples.
- Using sample period, extrapolate remaining edges
 - Now know first and last sample for each bit
 - Choose "middle" sample to determine message bit
- Can't go too long without a clock edge → 8b10b encoding

Serial, Point-to-Point Communications

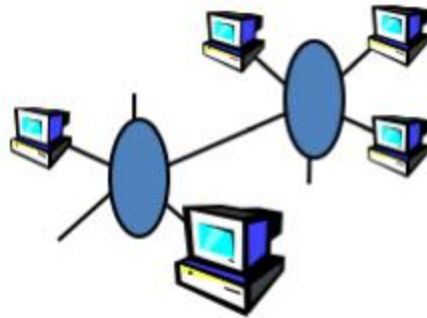
ETHERNET: Broadcast technology

- Sharing (contention) issues
- Multiple-drop-point issues...
- *bit-serial* (single wire!)
- “Packets” for multi-bit data



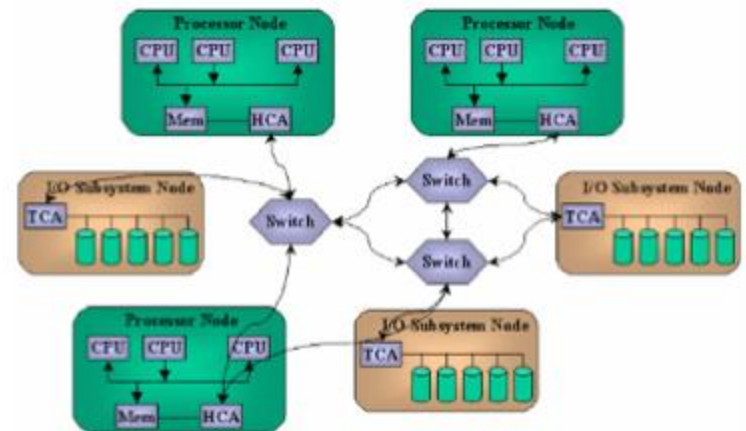
Evolution: Point-to-point

- 10BaseT, separate R & T wires
- Each link connects only 2 hosts, one sends, the other receives
- Network riddled with switches, routers



Serial point-to-point bus replacements

- Multi Gbit/sec serial links!
- PCIe, Infiniband, SATA, USB, ...
- Packets, headers
- Switches, routing
- Trend: localized, superfast, serial networks!

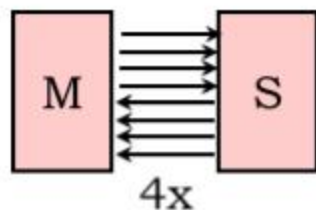
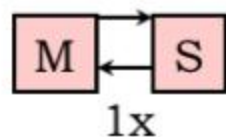
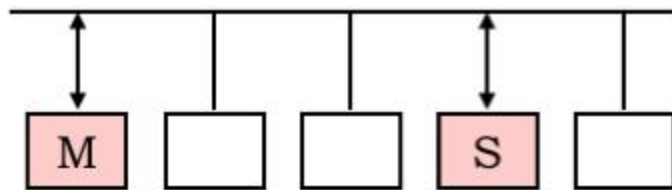


Improving on the bus:

lessons learned from the network world

Bus issues:

- shared medium → arbitrate between requesters
- clock skew → parallel bit lines, variable timings
- multiple masters → turnaround time
- impedance discontinuities, stubs → reflections



REPLACEMENT: fast unidirectional **serial point-to-point link**

- one transmitter, one receiver → no arbitration, no turnaround
- serial packets replace parallel wire bundles
- clock recovered from data bits → no skew problems
- unidirectional, point-to-point → good signal quality
- need more throughput? → use multiple serial links in parallel...
- need many-to-many communication? → switches (like Ethernet)
- complex interface → Moore's law to the rescue!

Communications in Today's Computers

QuickPath Interconnect:

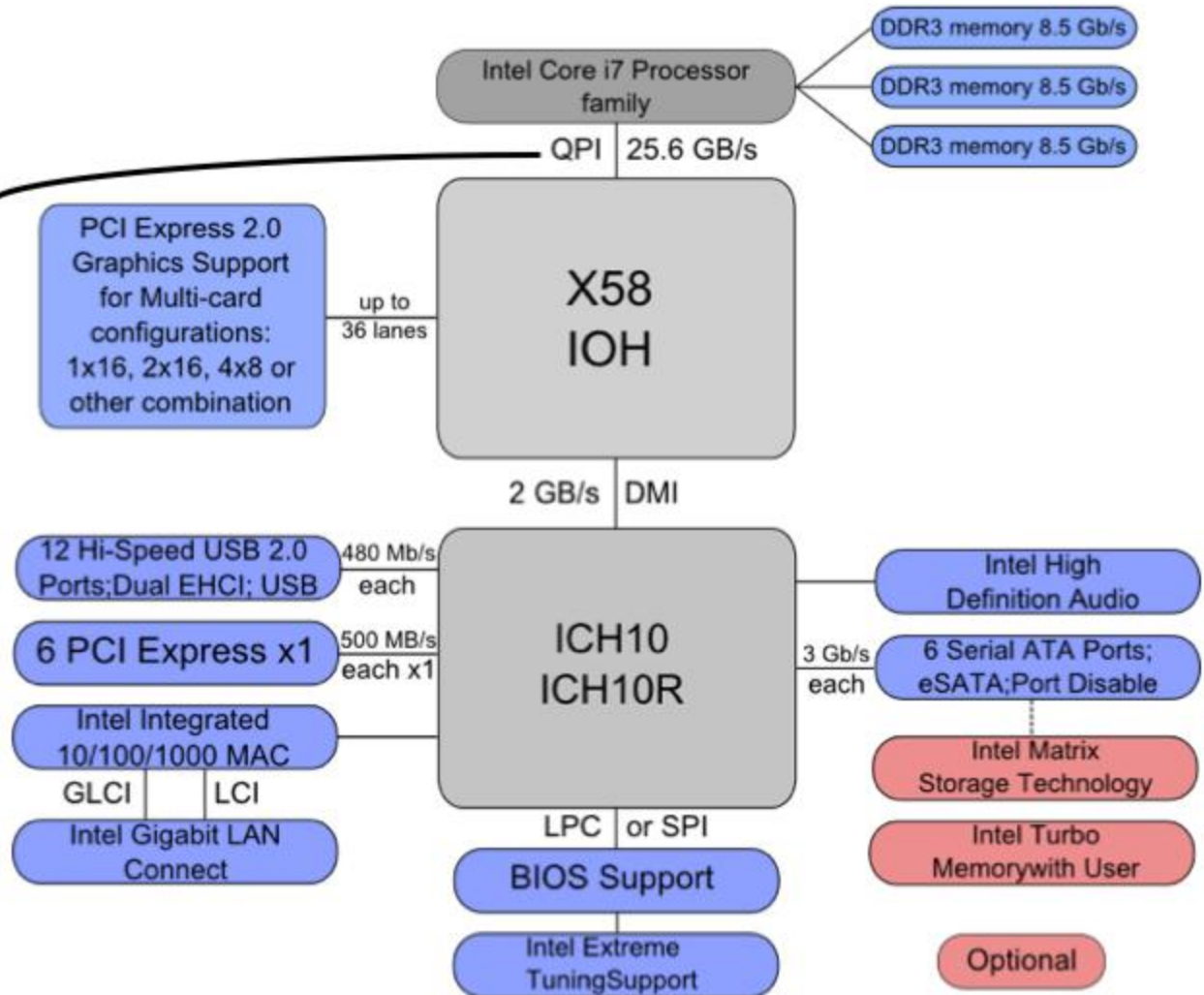
→ 20 data + clk

← 20 data + clk

Differential signaling

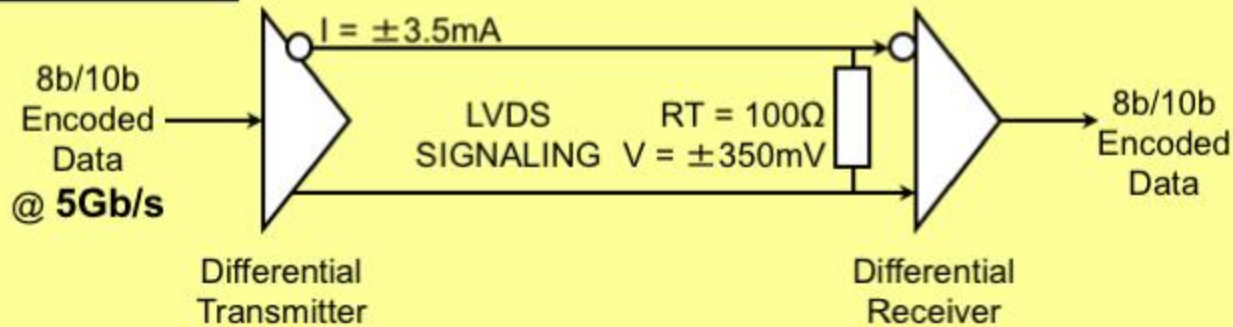
6.4 GT/s

*One bus to rule them all,
One bus to join them,
One bus to bring them all
And to the CPU bind them.*

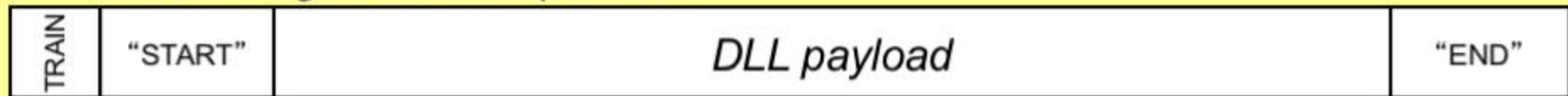


Example serial link: PCI Express (PCIe)

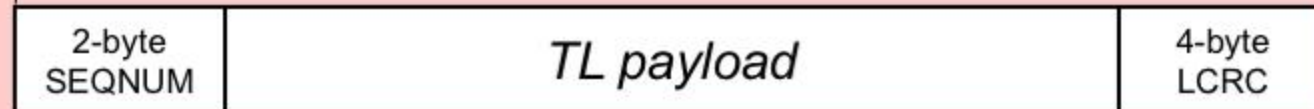
Physical Layer (v2.0)



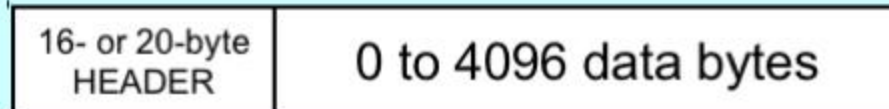
- PCIe x1: one differential pair in each direction, also x2, x4, x8, ..., x32
- Data is organized into packets:



Data Link Layer



Transaction Layer



Communication Topologies

asymptotic cost/performance tradeoffs

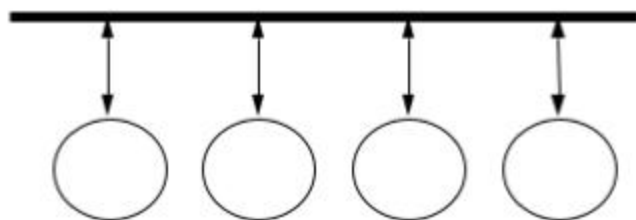
Goal: enable communications between n components

- Each point-to-point link requires one hardware unit.
- Each point-to-point communication requires one time unit.
- Each link operates independently

1-dimensional approaches:

BUS

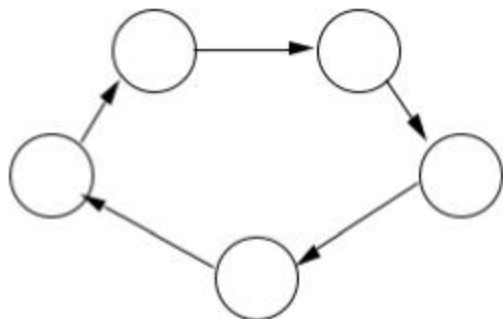
Shared communication channel allows only one message at a time



Throughput	$O(1)$
------------	--------

Latency	$O(1)$
---------	--------

Cost	$O(n)$
------	--------



RING

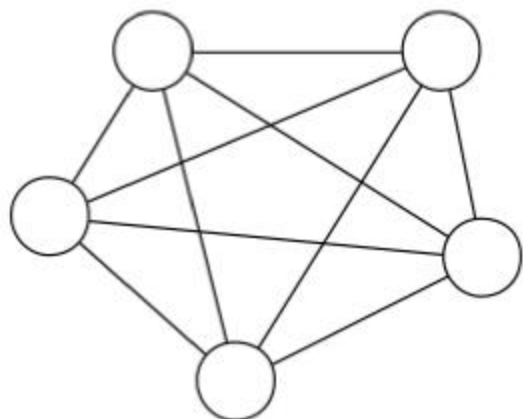
Each component has link to next component on ring

Throughput	$O(n)$
------------	--------

Latency	$O(n)$
---------	--------

Cost	$O(n)$
------	--------

Quadratic-cost Topologies



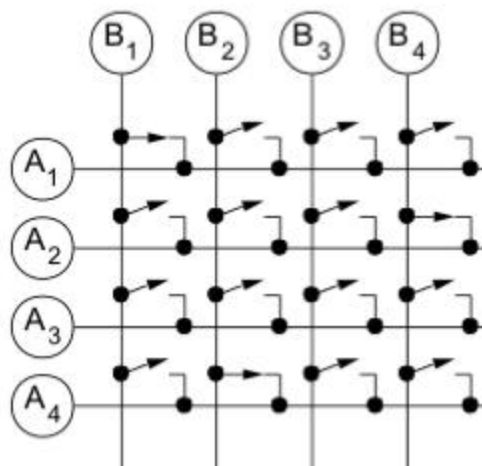
COMPLETE GRAPH

Dedicated lines connecting each pair of communicating nodes. There are $\sum_{i=1}^N (N - i) = O(n^2)$ links.

Throughput	$O(n^2)$
Latency	$O(1)$
Cost	$O(n^2)$

CROSSBAR SWITCH

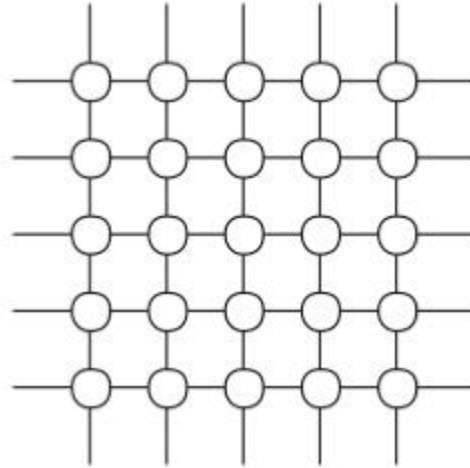
- Switch dedicated between each pair of nodes
- Each A_i can be connected to one B_j at any time
- Special cases:
 - A = processors, B = memories
 - A, B same type of node
 - A, B same nodes (complete graph)



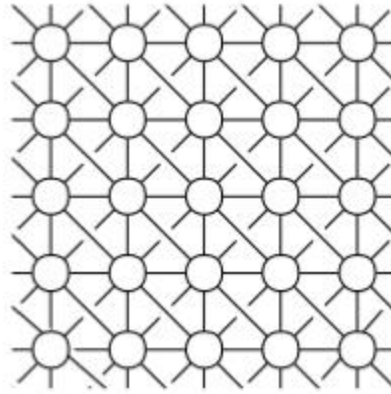
Throughput	$O(n)$
Latency	$O(1)$
Cost	$O(n^2)$

Mesh Topologies

2-Dimensional Meshes



4-Neighbor



8-Neighbor

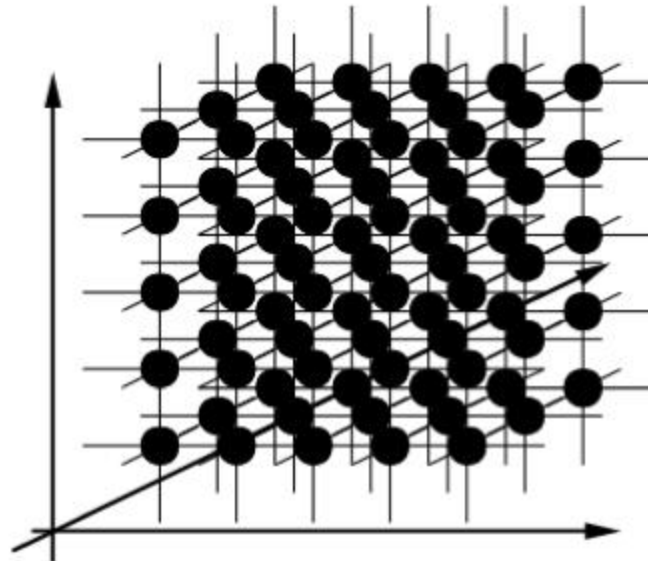
Throughput	$O(n)$
Latency	$O(\sqrt{n})$
Cost	$O(n)$

Nearest-neighbor connectivity:
Point-to-point interconnect

- minimizes delays
- minimizes “analog” effects

Store-and-forward

(some overhead associated with communication routing)



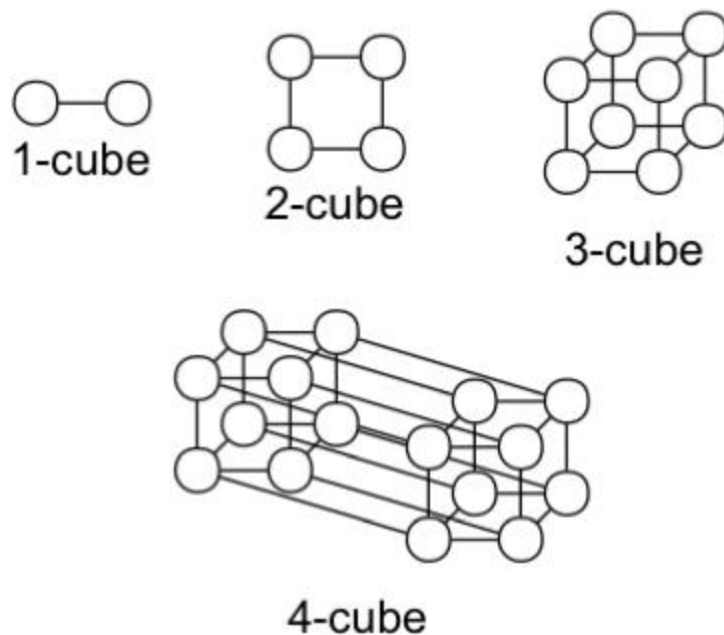
3-D, 6-Neighbor Mesh

Throughput	$O(n)$
Latency	$O(\sqrt[3]{n})$
Cost	$O(n)$

Logarithmic-latency Networks

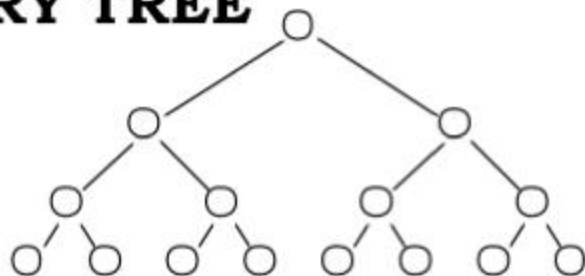
HYPERCUBE

D dimensions $\rightarrow 2^D$ nodes
Each node has D links



Throughput	$O(n \log_D n)$
Latency	$O(\log_D n)$
Cost	$O(n \log_D n)$

BINARY TREE



Throughput	$O(n)$
Latency	$O(\log_2 n)$
Cost	$O(n)$

Communication Technologies: Latency

- Theorist's view:
 - Each point-to-point link requires one hardware unit.
 - Each point-to-point communication requires one time unit.

Topology	\$	Theoretical Latency	Actual Latency
Complete graph	$O(n^2)$	$O(1)$	$O(\sqrt[3]{n})$
Crossbar	$O(n^2)$	$O(1)$	$O(n)$
1D Bus	$O(n)$	$O(1)$	$O(n)$
2D Mesh	$O(n)$	$O(\sqrt{n})$	$O(\sqrt{n})$
3D Mesh	$O(n)$	$O(\sqrt[3]{n})$	$O(\sqrt[3]{n})$
Tree	$O(n)$	$O(\log_2 n)$	$O(\sqrt[3]{n})$
N-cube	$O(n \log_D n)$	$O(\log_D n)$	$O(\sqrt[3]{n})$

- Engineer's view:
 - Loading increases with number of connections (bus, crossbar)
 - Nodes have size: limits possible 2D, 3D density (other topologies)

Communications Futures

Backplane buses have evolved into point-to-point links

- + links operate independently
- + links can be managed in groups
- + packetized data deals with errors

Specialized buses for memory

Networked “peripherals” for mobile devices...

New-generation communications...

- how should 100 (1000?) cores communicate?

