18
Buses

18-548/15-548 Memory System Architecture
Philip Koopman
November 11, 1998

Required Reading:	**Cragon 2.2.8**
Supplemental Reading:	**Hennessy & Patterson 6.3**
	Siewiorek & Koopman Appendix A
	Gustavson, Computer Buses - A tutorial
	Borrill comparison
	USB overview slides (on-line)
	PCI technical briefing (on-line)

Carnegie Mellon

Assignments

- **By November 18 about Multiprocessor Coherence:**
  - Cragon Chapter 4
  - Supplemental reading:
    - Hennessy & Patterson Chapter 8
    - Adve tutorial
    - Lenoski paper
    - Schimmel: pp. 59-68, 83-87, 99-104
- **Homework 10 due November 18**
- **Lab #5 due Friday November 20**

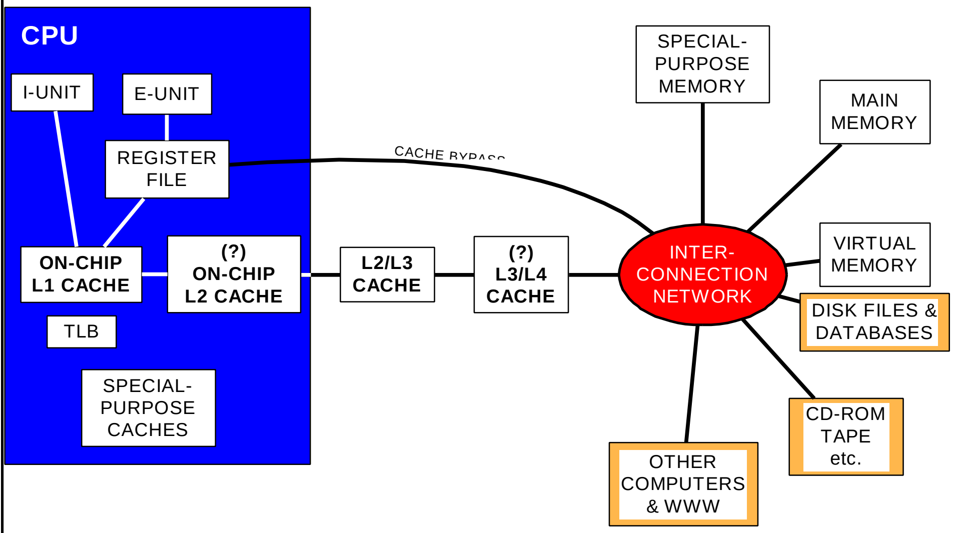
Where Are We Now?

- **Where we've been:**
  - Cache memory
  - Main memory
  - Vector computing
- **Where we're going today:**
  - Buses -- how do you connect everything?
- **Where we're going next:**
  - Multiprocessor Coherence
  - Fault tolerance in the memory hierarchy (error detection/correction, etc.)
  - Test #3

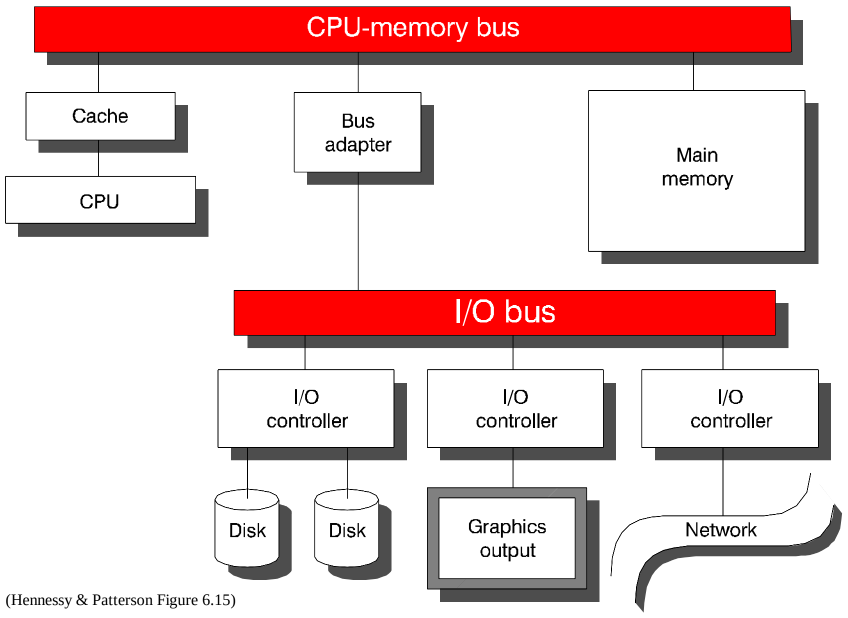
Preview

- **Interconnection networks trade cost, bandwidth, latency**
  - Various cost/performance points in interconnection design space
- **IBM PC family**
  - ISA bus operation
  - Evolution over time
- **Buses also have cost, bandwidth, latency tradeoffs**
  - Parallel buses
  - Serial buses

Interconnection Network



"Internal" Computer Buses



(Hennessy & Patterson Figure 6.15)

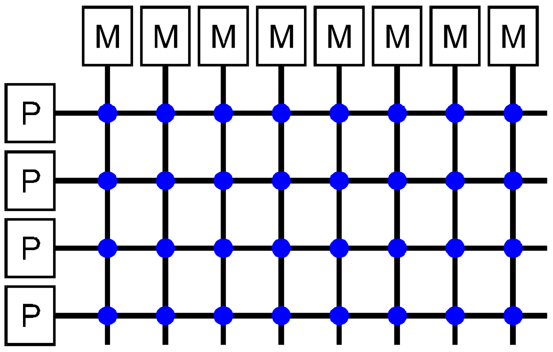
GENERAL INTERCONNECTION

Ad-Hoc Point-to-Point Interconnection

- **Wires from every subsystem to every other subsystem**
  - Highest bandwidth
  - High system cost
    - Connector costs or pin costs
    - Combinatorial explosion as number of subsystems grows
    - Problems with designing for expandability
  - In older designs most bandwidth goes unused
    - ... so, use some sort of regular structure instead
- **Scales poorly**
  - Wires for every path
  - Connector on each node for every connection

Crossbar -- Generalized Point-to-Point

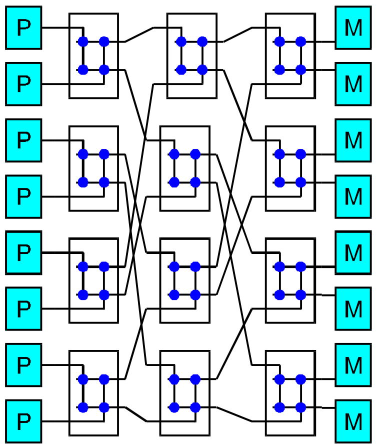
- **Crossbar switch permits connecting, for example, *n* CPUs to *m* memory banks for simultaneous accesses**
  - Cost is *n*m* switches
  - Latency is a single switch delay
- **Used for high-bandwidth with few resources**
  - Connecting a few processors to interleaved memory
  - Vector Register File to Vector Data Path
- **Scales poorly for large n or m**



Crossbar Switch

Multi-Stage Interconnect

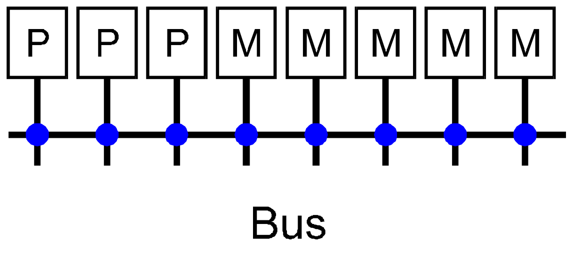
- **Omega network provides potentially high bandwidth, but suffers from blocking/congestion**
  - Each “hop” on network requires passing through an embedded processor/switch
  - Omega network provides high potential bandwidth, but at cost of latency of log *n* switching stages = O(*n* log *n*) switches
  - Other topologies possible, but all involve a cost *vs.* bandwidth tradeoff



Omega Network

"Bus" Interconnect

- **Single communication resource shared by system**
  - Minimum cost for connections & switches
  - Minimum latency if the interconnection resource is idle
  - Cost is O(n) connections
- **Concurrency & replication can still be exploited**
  - Width of bus
  - Pipelining of bus access



Alternate Approaches

- **"Star" configuration**
  - Single hub node with multiple nodes connected with a single node in the middle
  - Potential congestion at hub node unless it is high bandwidth (*e.g.* ATM switch)
- **"Tree" configuration**
  - Binary tree has two children for every parent; half the nodes are leaf nodes
  - Can suffer congestion at root node
- **"Mesh" connection**
  - Nearest neighbor connection (*e.g.* N.E.W.S. network on SIMD machines)
  - High bandwidth, no bottlenecks, high point-to-point latency
- **"Hypercube" connection**
  - All nodes differing by 1 bit in address are connected
  - Potential bottlenecks on "long" connections unless algorithm maps well
  - Hypercube is a superset of a mesh, but with "express" connections

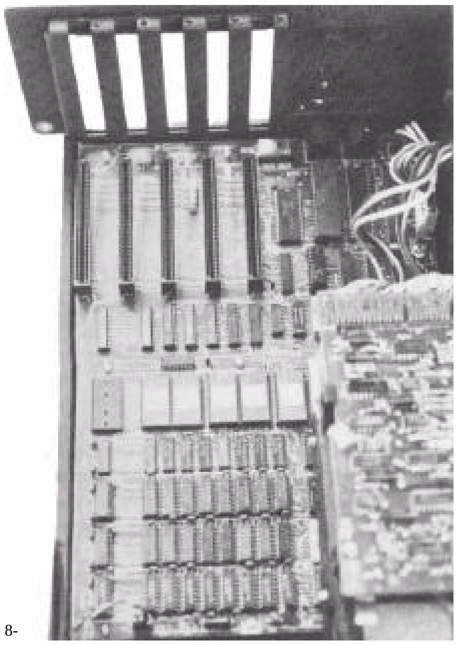
IBM PC Family
Bus Organization

PC Family System Bus Evolution

- **ISA -- original IBM PC & PC-XT**
  - 62 contacts, 8 bit data bus, synchronous operation
  - Memory & I/O on single bus
  - Split address & data; 3 DMA channels, 6 IRQ lines
  - Separate I/O & memory address spaces
  - Single Master
  - 4.77 MHz original operation (same as CPU clock)
- **AT Bus -- adds second socket to extend to 16 bits**
  - 62 + 36 = 98 contacts for 16-bit extension of ISA (backward compatible)
    +8 data bits, +7 address bits, + 3 DMA channels, + 5 IRQ lines
  - Multi-master
  - Operation up to 8.33 MHz

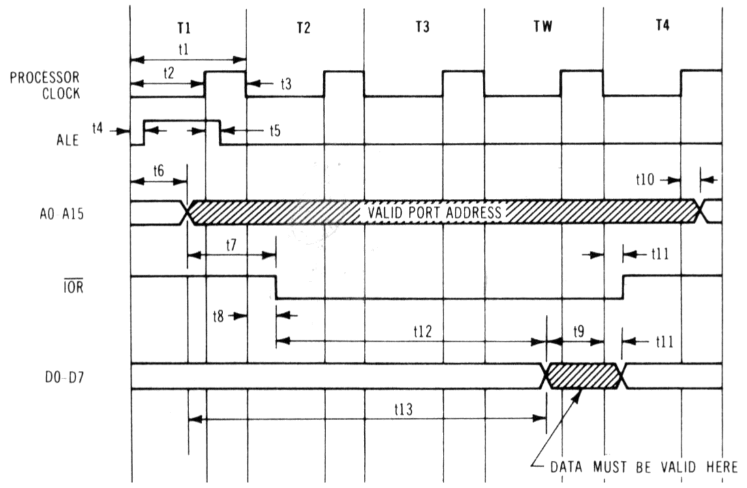
Original ISA Bus Pinout

“CHIP” SIDE	“SOLDER” SIDE
A1: IOCHK#	B1: GND
A2: SD7	B2: RESETDRV#
A3: SD6	B3: +5V
A4: SD5	B4: IRQ2
A5: SD4	B5: -5V
A6: SD3	B6: DRQ2
A7: SD2	B7: -12V
A8: SD1	B8: (unused)
A9: SD0	B9: +12V
A10: IOCHRDY	B10: GND
A11: AEN	B11: SMEMW#
A12: SA19	B12: SMEMR#
A13: SA18	B13: IOW#
A14: SA17	B14: IOR#
A15: SA16	B15: DACK3#
A16: SA15	B16: DRQ3
A17: SA14	B17: DACK1#
A18: SA13	B18: DRQ1
A19: SA12	B19: REFRESH#=DACK0#
A20: SA11	B20: BCLK (4.77 MHz)
A21: SA10	B21: IRQ7
A22: SA9	B22: IRQ6
A23: SA8	B23: IRQ5
A24: SA7	B24: IRQ4
A25: SA6	B25: IRQ3
A26: SA5	B26: DACK2#
A27: SA4	B27: TC
A28: SA3	B28: BALE
A29: SA2	B29: +5
A30: SA1	B30: OSC (14.3 MHz)
A31: SA0	B31: GND


(Eggebrecht Figure 8-1)

ISA Bus Operation

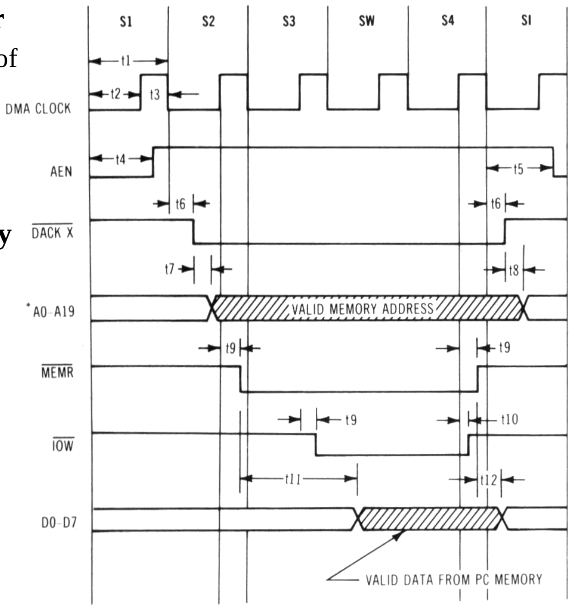
- **5 clock operation for I/O Read (4 clocks for memory read; same idea)**
  - Operations synchronized with processor clock; no handshake from I/O card
  - In practice, IOR may be used as an asynchronous signal to put data on the bus



(Eggebrecht Figure 6-2)

ISA Direct Memory Access (DMA) Operation

- **Separate DMA controller**
  - Counter to track number of words remaining
  - "Cycle steals" bus bandwidth, transparent to programs
- **Data moves from memory to I/O**
  - I/O card asserts DRQx
  - I/O eventually receives DACKx from DMA controller
  - DMA controller asserts MEMR and IOW to accomplish a concurrent memory read and I/O write operation



(Eggebrecht Figure 6-

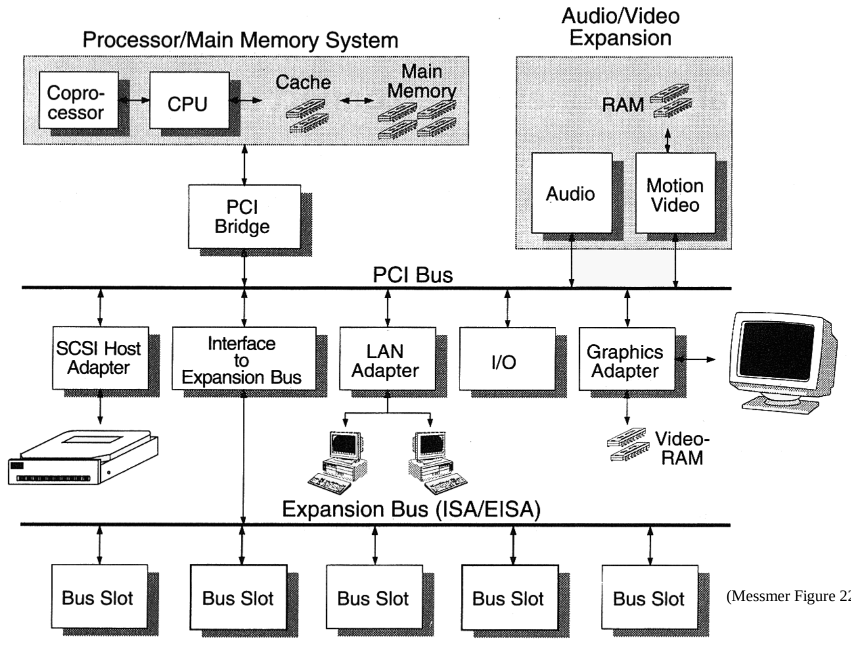
32-Bit PC System Buses

- **EISA Bus -- evolutionary extension to 32 bits**
  - Two contact layers (special connector for ISA compatibility)
    - 98 contact AT BUS compatible layer
    - 90 contact stacked EISA layer
  - Can transfer 64-bit data by multiplexing address lines
  - 8.33 MHz operation
- **Microchannel (MCA) -- IBM proprietary bus**
  - PS/2 & System/6000 -- attempt to capture PC market share (didn't work)
  - 294 contacts on a fine-pitch connector
  - 10 MHz synchronous transfers
  - Supports asynchronous transfers ~7% bandwidth improvement
    - Improvement because some transfers can be accomplished in less than an integral number of clock cycles

PCI Bus

- **Peripheral Component Interconnect**
  - Recognizes split between memory function & I/O function
  - Up to 33 MHz operation
  - 32-bit bus @ 133 MB/sec with 124 contacts
  - Expansion to 64-bit bus with 124+64 =188 contacts
- **PCI Bridge used**
  - Connects main memory bus to PCI bus
  - Can also have a bridge to an "expansion bus" (ISA/EISA)

PCI Bus System Architecture

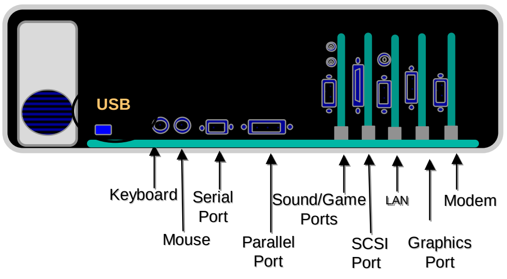


(Messmer Figure 22.1)

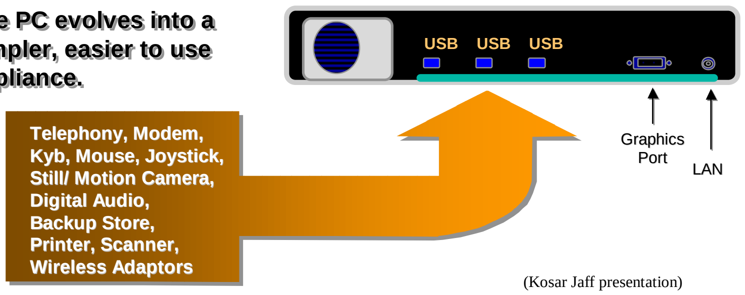
PC External Peripheral Bus

- USB = Universal Serial Bus

USB in 1996:
Initially introduced as an incremental connector for new applications.



USB Future:
The PC evolves into a simpler, easier to use appliance.


USB
USB
USB
Graphics Port
LAN
Telephony, Modem, Kyb, Mouse, Joystick, Still/ Motion Camera, Digital Audio, Backup Store, Printer, Scanner, Wireless Adaptors

(Kosar Jaff presentation)

BUS DESIGN & PERFORMANCE

Bus

- **Bus is a shared system interconnection**
  - CPU to cache
  - Cache to main memory
  - CPU or main memory to I/O
  - Typically high-speed connection, and often carries processor/memory traffic
  - Typically accessed transparently via a memory address
- **As bandwidth demands on system increase, number of buses increases**
  - "Back-side" cache bus
  - Separated memory and I/O bus
  - Separated external peripheral & internal peripheral bus
- **Bus width is the primary cost/speed tradeoff**
  - Parallel buses transmit multiple bits at a time
  - Serial buses transmit one bit at a time

Synchronous vs. Asynchronous

- **Synchronous bus uses clock signal for data timing**
  - Parallel buses have a separate "clock" signal
  - Serial buses synchronize periodically (e.g., every byte) and use stable time bases at transmitter/receiver (*e.g.,* UART operation)
  - Good for short, high-speed buses
  - Concentrates all the design headaches into two places
    - Clean, stable clock
    - State machines must know how many clocks each operation takes
- **Asynchronous bus uses handshake signals/data waveforms for timing**
  - Example handshake: "I want data" -- "Here is the data" -- "OK, I got it"
    - Prevalent in older computers, especially with very slow I/O devices
    - Good for long, slower speed buses where handshakes propagate with data
    - Can lead to use of "one-shots", which are Bad
  - Example timing: 1 = HHL 0 = HLL ("Kansas City Standard" for audio tape)
    - Good for serial data streams over long lengths, or on tape (with phase distortion)
    - Less bandwidth efficient than synchronous data streams

Parallel Bus Design Options

- **Parallel bus transmits multiple bits of data concurrently**
  - High performance = expensive, “big”, complex
  - Low cost = inexpensive, “small”, simple

Option	High Performance	Low Cost
Bus Width	Separate address & data lines	Multiplexed address & data
Data Width	Wider is faster (e.g., 256 bits)	Narrower is cheaper (e.g., 8 bits)
Transfer Size	Burst transfers for reduced overhead	Single-word transfers for simplicity & low latency
Bus Masters	Multi-master (requires arbitration)	Single master (CPU can be bottleneck)
Pipelining	Split address & data transactions (packet switched)	Continuous connection (circuit switched)
Clocking	Synchronous	Asynchronous
Logical View	Memory mapped	Message-based

(Adapted from Hennessy & Patterson Figure 6.9)

Parallel Bus Performance

- **Bandwidth -- limited by cost and transmission line effects**
  - 64-bit or 128-bit data bus common (but, fewer bits on cost-sensitive systems)
    - Why was the 8088 used instead of the 8086 in the original IBM PC?
  - Bus speed often limited to 50 - 66 MHz due to transmission line effects
  - Up to 528 MB/sec for 64-bit bus at 66 MHz

- **Latency -- limited by distance and need for drivers**
  - Multiple clock latency, but can pipeline and achieve 1 clock/datum throughput
  - (Be careful about “bus clocks” *vs.* “processor clocks”)

  - Also determined by waiting for other transactions to complete...

Serial Bus Design Options

- **Serial bus transmits a single bit of data at a time**
  - High performance = expensive, “big”, complex
  - Low cost = inexpensive, “small”, simple

Option	High Performance	Low Cost
Bus Speed	Bit time faster than 2* t_{pd}	Bit time slower than 2* t_{pd}
Transfer Size	Large messages for reduced overhead	Small messages for simplicity & low latency
Bus Masters	Multi-master (requires arbitration)	Single master (CPU can be bottleneck)
Pipelining	Split command & response transactions (packet switched)	Continuous connection (circuit switched)
Clocking	Synchronous (requires stable time base)	Asynchronous

Serial Bus Performance

- **Bandwidth -- limited by component speeds (within loose cost constraints)**
  - With only 1 bit of connectivity, can afford a relatively expensive connector & transceiver
  - In high speed applications can shift to multiple bits on medium concurrently, treat as a real transmission line

- **Latency -- limited by message length & protocol overhead**
  - Number of bits in message determines part of latency to send data
  - Overhead for arbitration, error correction, routing determines rest of latency

  - Also determined by waiting for other transactions to complete...

REVIEW

Review

- **Interconnection networks trade cost, bandwidth, latency**
  - Crossbars -- expensive, big, fast
  - Multistage -- less expensive, less big, less fast
  - Bus -- least expensive, small, fast
- **Buses also have cost, bandwidth, latency tradeoffs**
- **Hierarchy of buses becoming prevalent**
  - Buses close to CPU -- cache, memory
  - “In-the-box” buses -- disks, I/O controllers
  - External peripheral buses -- keyboard, audio, video, printing
  - Inter-computer connectivity -- communication networks