Performance of system

Lecture 11

The Bus

- The CPU communicates with the other components via a bus.
  - A *bus* is a set of wires that acts as a shared but common data path to connect multiple subsystems within the system.
  - It consists of multiple lines, allowing the parallel movement of bits.
- Buses are low cost but very versatile, and they make it easy to connect new devices to each other and to the system.
  - At any one time, only one device (a register, the ALU, memory, or some other component) may use the bus.
    - sharing often results in a communications bottleneck.

Bus

- The Data Bus
  - Used for moving data between sub systems i.e. data to/from main memory /IO/ processor
  - Each line can carry 1 bit
    - Number of lines determine how many bits can be transferred at one time
    - Number of lines in data bus are called width of data bus
  - Width of data Bus effects system performance
    - If data bus is 8 bits wide & instruction is 32 bits or 4 bytes long then how many fetch cycles are required?
- Width of P-IV data bus is 64 Bits

Bus

- Address Bus
  - Address lines are generally used to address memory location or I/O ports
  - It is used to designate source or destination of data on data bus
  - Address bus width effect the memory size
    - It determines Max possible memory capacity of system
  - P-IV address buses size is 32 Bits
- Data & address bus are shared by all components
  - Sharing demands the availability of protocol is important
  - There must be a mean of controlling their use

Control Lines

- Control lines are used to control the access/use of data/address bus
  - Control bus indicate which device has the permission to use the bus and for what purpose
    - Control Signals indicate specific operation to be performed
  - Also transfer acknowledgement for bus request, interrupt and synchronization signals
- Typical control lines are memory write, memory read, I.O write, I.O read, Bus request, Bus grant etc

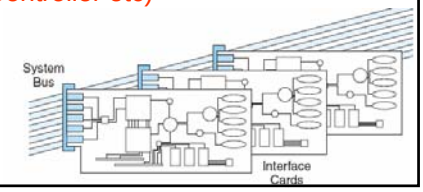
Bus Types

- As buses transport different types of information, and different types of devices using the bus, the bus itself is of different types
  - Processor – Memory bus (very short, very high speed)
  - I/O buses (longer than processor – memory)
  - Backplane bus (motherboard)
- Bus hierarchy is also used

Bus Types

- Processor – Memory bus (very short, very high speed)
  - usually designed for specific processor.
  - highest bandwidth , Highest Speed
- I/O buses (longer than processor – memory)
  - Standard buses used in different computers
  - Other devices are connected to it (disk, printer etc)
  - Lowest bandwidth , Lowest Speed
- Backplane bus
  - backplane = an interconnection structure within the chassis
  - connects processor & memory to nearby components (multimedia accelerators, network cards, I/O Controller etc)

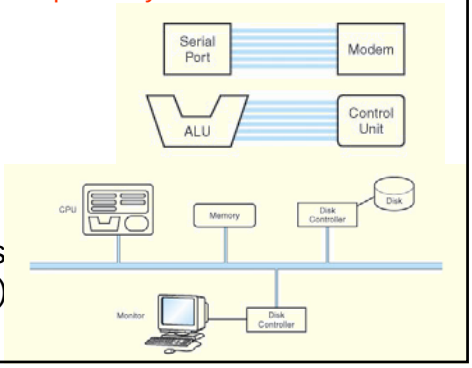
- Bus hierarchy is also used



The Bus

- The speed of the bus is affected by its length as well as by the number of devices sharing it.
  - Quite often, devices are divided into *master* and *slave* categories,
    - A master device is one that initiates actions and
    - A slave is one that responds to requests by a master.

A bus can be *point-to-point*, connecting two specific components or it can be a *common pathway* that connects a number of devices, requiring these devices to share the bus (*multipoint* bus)



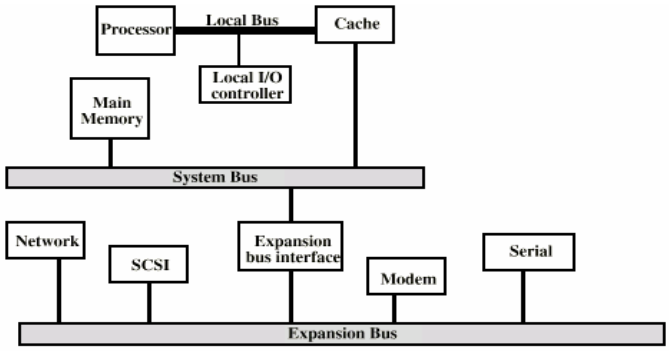
The Bus

- In a master-slave configuration, where more than one device can be the bus master, concurrent bus master requests must be arbitrated.
- Four categories of bus arbitration are:

- **Daisy chain:**
  - Permissions are passed from the highest-priority device to the lowest.
    - "starved out", simple but not fair.
- **Centralized parallel:**
  - Each device is directly connected to an arbitration circuit.
    - Bottlenecks can result
- **Distributed using self-detection:**
  - Devices decide which gets the bus among themselves.
    - Devices not the central arbiter
- **Distributed using collision-detection:**
  - Any device can try to use the bus. If its data collides with the data of another device, it tries again.
    - Ethernet

Single Bus Problems

- Lots of devices on one bus leads to:
  - Propagation delays
    - Long data paths mean that co-ordination of bus use can adversely affect performance
- Most systems use multiple buses to overcome these problems

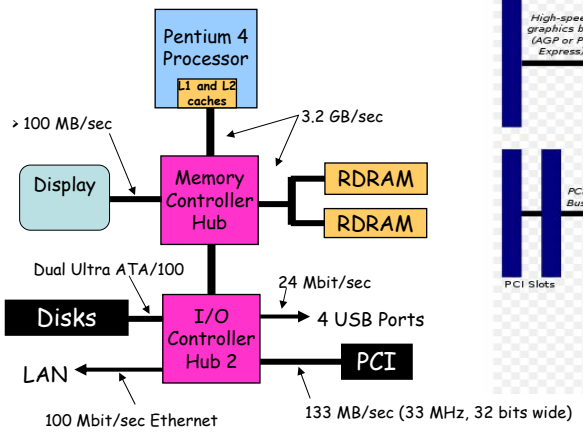


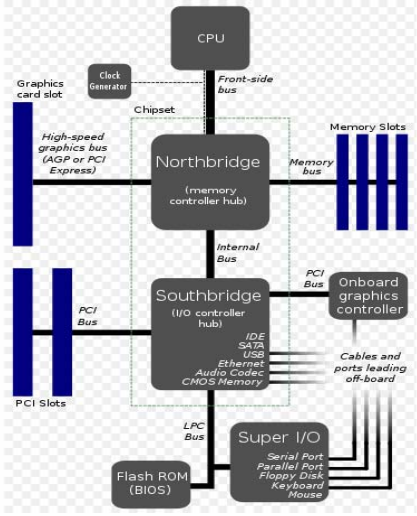
Bus

- Most PC also have a local bus
  - Data bus that connects a peripheral device directly to the CPU
  - More fast
    - The significance of direct connection to the CPU is avoiding the bottleneck created by the expansion bus, thus providing fast throughput.
      - Expansion bus: A collection of wires and protocols that allows the expansion of a computer by inserting *expansion boards*
      - Traditionally, PCs have utilized an expansion bus called the *ISA bus*.
        - » ISA: ***I**ndustry **S**tandard **A**rchitecture bus*, the bus architecture used in PC/AT.
      - SCSI, **Small Computer System Interface**

Front side Bus

- Carries data between the CPU and the Northbridge.





Bus in P IV

Types

- Synchronous Bus: (Processor–Memory)
  - Occurrence of event on the bus is coordinated by clock
  - Advantage:
    - Can run very fast
  - Disadvantages:
    - Devices on the bus must run at the same clock rate
    - Clock skew (clock signal sent from the clock circuit arrives at different components at different times) effects bus length
      - Bus must be kept as short as possible

Types

- Asynchronous Bus: (I/O)
  - To ensure timings, control bus coordinate the operations through handshake protocol
    - Occurrence of event on the bus depends upon occurrence of previous event and additional control lines (ReadReq, Ack, DataRdy etc) are required
  - Advantage:
    - It can accommodate a wide range of devices
  - Disadvantage:
    - Complex communication protocol is needed

Performance Equation

- Clock speed should not be confused with CPU performance.
- The CPU time required to run a program is given by the general performance equation:

$$\text{CPU Time} = \frac{\text{seconds}}{\text{program}} = \frac{\text{instructions}}{\text{program}} \times \frac{\text{avg. cycles}}{\text{instruction}} \times \frac{\text{seconds}}{\text{cycle}}$$

  - We can improve CPU throughput when we:
    - reduce the number of instructions in a program,
    - reduce the number of cycles per instruction, or
    - reduce the number of nanoseconds per clock cycle.

I/O subsystem

- A computer communicates with the outside world through its input/output (I/O) subsystem.
  - I/O is the transfer of data between primary memory and various I/O peripherals
- I/O devices connect to the CPU through various interfaces.
  - Devices not connected directly to the CPU but through an interface that handles the data transfer
  - CPU communicate with I/O devices through input/ output registers

I/O subsystem

- Exchange of data is performed in two ways
  - Memory mapped I/O
    - where the I/O device behaves like main memory from the CPU's point of view.
    - The registers in the interface appears in memory map
      - Advantage faster access, disadvantage memory space compromised
  - Isolated I/O
    - Or isolated /instruction-based, where the CPU has a specialized I/O instruction set.
      - Advantage saving to memory space, disadvantage requires specialized I/O instructions

I/O and Performance

- Sluggish I/O throughput can have a ripple effect, dragging down overall system performance.
  - The fastest processor in the world is of little use if it spends most of its time waiting for data.
  - If we really understand what's happening in a computer system we can make the best possible use of its resources.

Amdahl's Law

- The overall performance of a system is a result of the interaction of all of its components.
- System performance is most effectively improved when the performance of the most heavily used components is improved.
- This idea is quantified by Amdahl's Law:

$$S = \frac{1}{(1-f) + \frac{f}{k}}$$

where S is the overall speedup; f is the fraction of work performed by a faster component; and k is the speedup of the faster component.

Amdahl's Law

- Amdahl's Law gives us a handy way to estimate the performance improvement we can expect when we upgrade a system component.
  - On a large system, suppose we can upgrade a CPU to make it 150% faster for $10,000 or upgrade its disk drives for $7,000 to make them 250% faster.
    - Processes spend 70% of their time running in the CPU and 30% of their time waiting for disk service.
- An upgrade of which component would offer the greater benefit for the lesser cost?

Amdahl's Law

- The processor option offers a 130% speedup:

$$f = 0.70,\ k = 1.5 \qquad S = \frac{1}{(1 - 0.7) + 0.7/1.5}$$

- And the disk drive option gives a 122% speedup:

$$f = 0.30,\ k = 2.5 \qquad S = \frac{1}{(1 - 0.3) + 0.3/2.5}$$