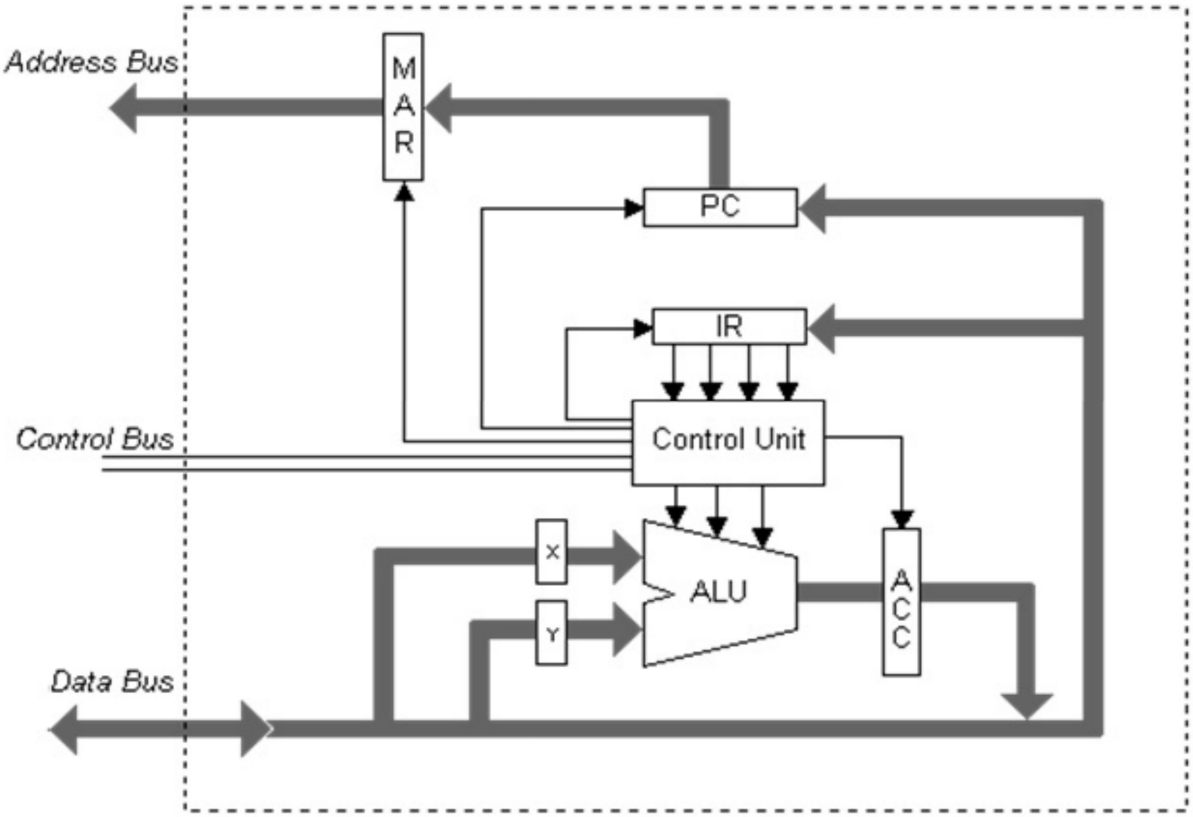
The Computer System Bus Model

- The bus model consider the computer parts as:
  - CPU : combines the ALU and the control unit into one functional unit
  - Memory
  - Input/Output (I/O)
  - System Bus

A simple CPU design



System Bus

- The system bus :is a shared pathway system made up of collections of wires that are grouped by function that mainly cares about the communications among the other components.
- only one device can send data while other devices simultaneously listen but only one device receives
- Devices
    - Master
    - Slave

Master	Slave	Example
CPU	Memory	Fetching Instructions/Data
CPU	I/O	Initiating data transfer
CPU	Coprocessor	Handing off floating point operation
I/O	Memory	Direct Memory Access (DMA)
Coprocessor	Memory	Fetching operands

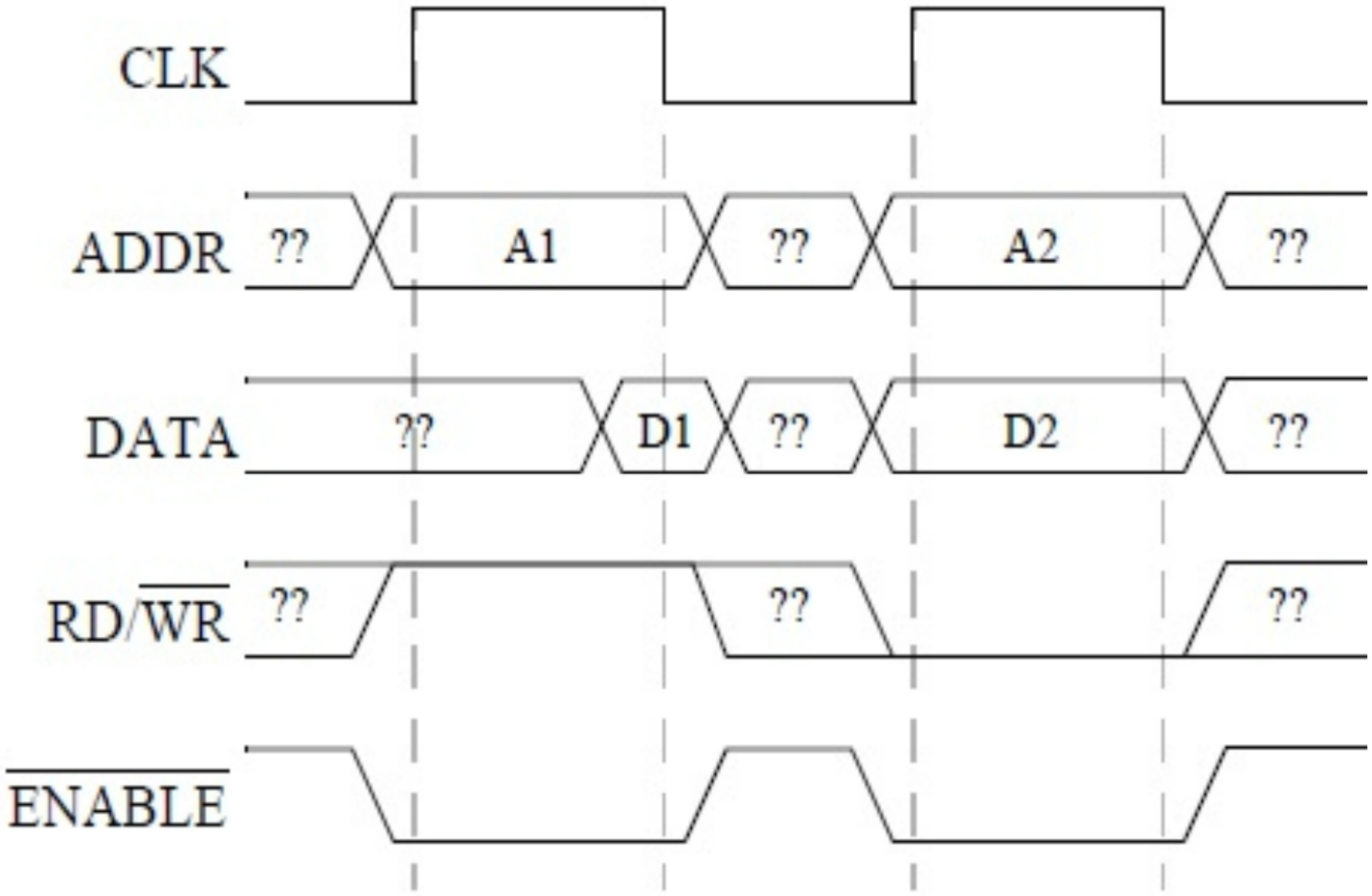
Bus Types

- Buses could be classified into
  - Timing
    - Synchronous
    - Semi-synchronous
    - Asynchronous
  - Data type
    - Data Bus
    - Address Bus
    - Control Bus
  - Dedicated or Multiplexed

Synchronous Bus

- Contains a clock (CLK) that sends out a sequence of 1's and 0's at timed intervals to synchronize bus operations.
- The clock signal is derived from the master system clock
- Master outputs are valid on the rising edge of CLK, and stay valid through the falling edge of CLK.
- Slave outputs (for read) is valid by the falling edge of CLK

Synchronous Bus



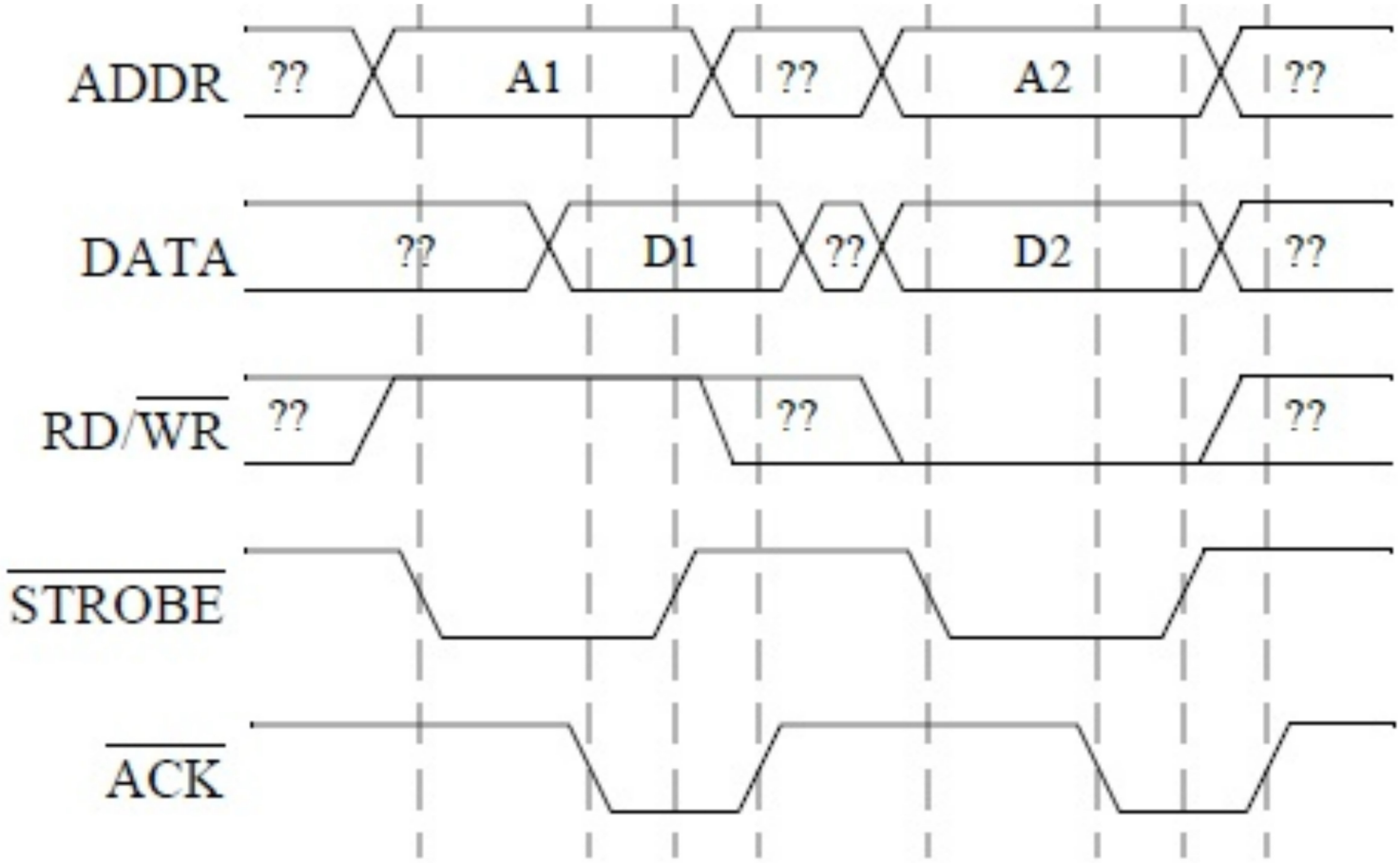
Semi-synchronous Bus

- Everything is synchronized to bus clock, but the transactions take variable number of cycles
- Slave asserts ACK to indicate that data is valid at clock edge.
- Extra clock cycles between start and end due to late ACK are called wait states

Asynchronous Bus

- No clock signal
- Transmitters and receivers in all timing based on control signal edges
- Handshaking protocol
    - STROBE: master to slave signal
        - slave reads on falling edge
    - ACK: slave to master signal
        - master reads on falling edge)

Asynchronous Bus



	Synchronous	Asynchronous
Advantages	Lower overhead and thus, greater throughput	• Simple, doesn't require synchronization of both communication sides • Cheap, because Asynchronous transmission require less hardware • Set-up is faster than other transmissions, so well suited for applications where messages are generated at irregular intervals, for example data entry from the keyboard and the speed depends on different applications.
Disadvantage	• Slightly more complex • Hardware is more expensive	• Large relative overhead, a high proportion of the transmitted bits are uniquely for control purposes and thus carry no useful information

Data Bus

- Exchanging data between the processor, memory and peripherals
- Bi-directional
- Bus Width: is the number of wires used in the data bus
  - E.G. : 32-bit data bus has 32 individual wires, each of which carries one bit of data

Address Bus

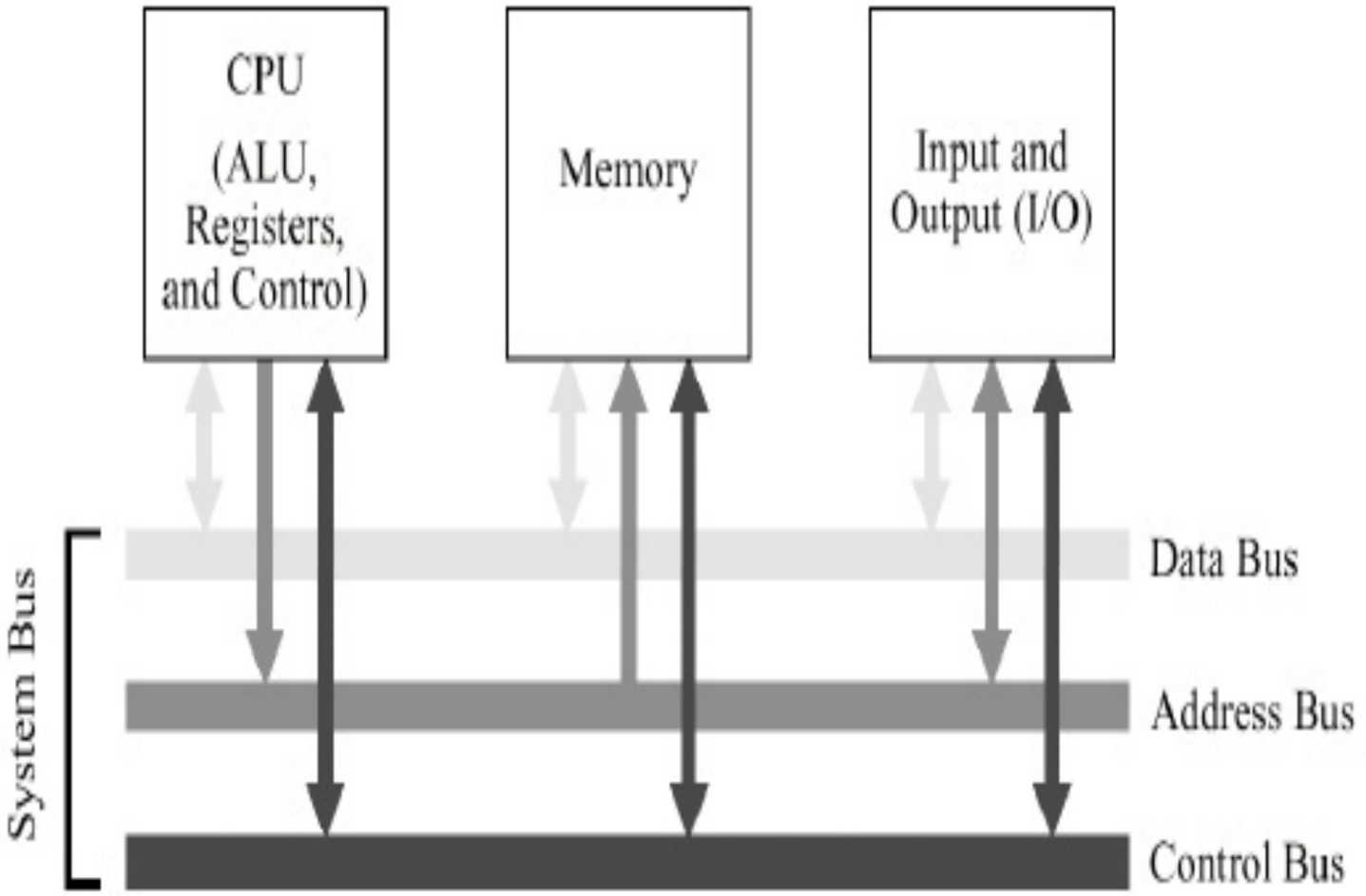
- Identifies where the information is being sent.
  - microprocessor and memory
- Width of the address bus corresponds to
  - maximum addressing capacity of the bus (2^lines)
- Addresses are transferred in binary format, with each line of the address bus carrying a single binary digit.

Control Bus

- Carries the signals from the **CU** relating to the control and co-ordination of activities across computer.
- Different architectures result in differing number of lines
  - Data In, Data Out
  - Data In/Out

Control Bus

- **Control buses indicates that :**
  - **Memory Write:** data on the bus to be written into the addressed location
  - **Memory Read:** data from the addressed location to be placed on the bus
  - **I/O Write:** data on the bus to be output to the addressed I/O port
  - **I/O Read:** data from the addressed I/O port to be placed on the bus
  - **Transfer ACK:** data have been accepted from or placed on the bus
  - **Bus request:** a module needs to gain control of the bus
  - **Bus grant:** a requesting module has been granted control of the bus
  - **Interrupt request:** Indicates that an interrupt is pending
  - **Interrupt ACK:** Ack that the pending interrupt has been recognized
  - **Clock:** Is used to synchronize operations
  - **Reset:** Initializes all modules

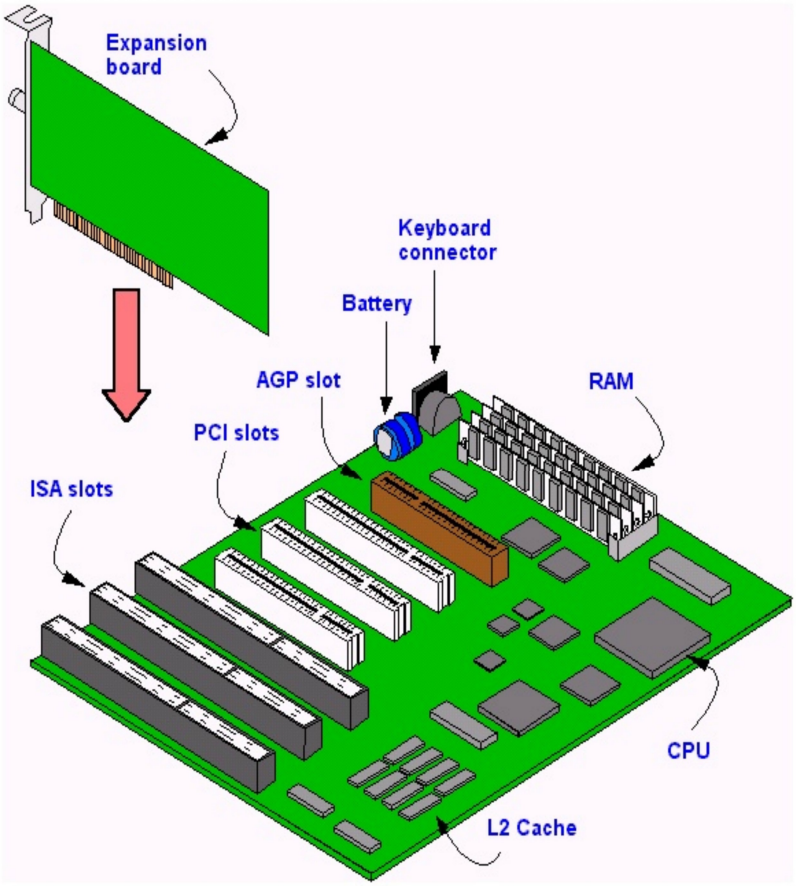

CPU
(ALU,
Registers,
and Control)
Memory
Input and
Output (I/O)
System Bus
Data Bus
Address Bus
Control Bus

Dedicated & Multiplexed

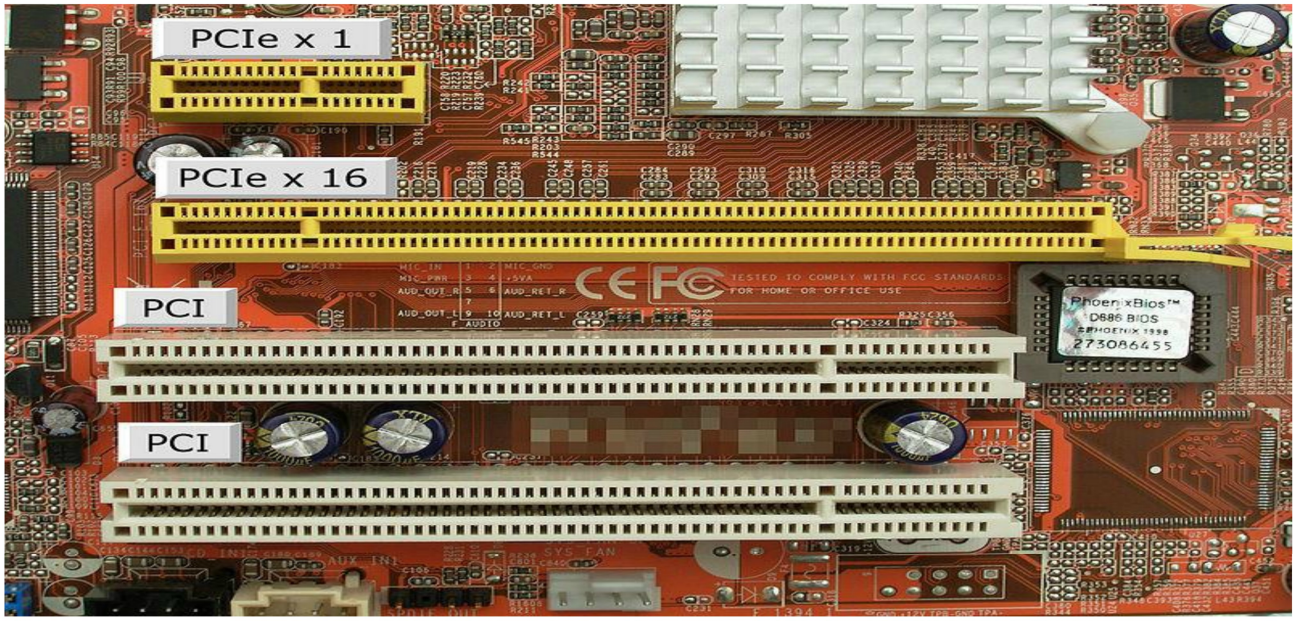
- Dedicated : bus used only for one purpose
  - E.G. Data bus, Address bus
- Multiplexed : bus using the same lines at different times for multiple purposes .
  - Require fewer lines and chips can be limited in the number of pins that can be physically attached.
  - Data and addresses may appear on the bus at different times.

Industry Standard Architecture (ISA)

- ISA was introduced by IBM.
- ISA
  - –8-bit computer bus
  - –expanded to a 16-bit bus in 1984.
- In 1993, Intel and Microsoft introduced a PnP ISA bus that allowed the computer to automatically detect and setup computer ISA peripherals, such as a modem or sound card.

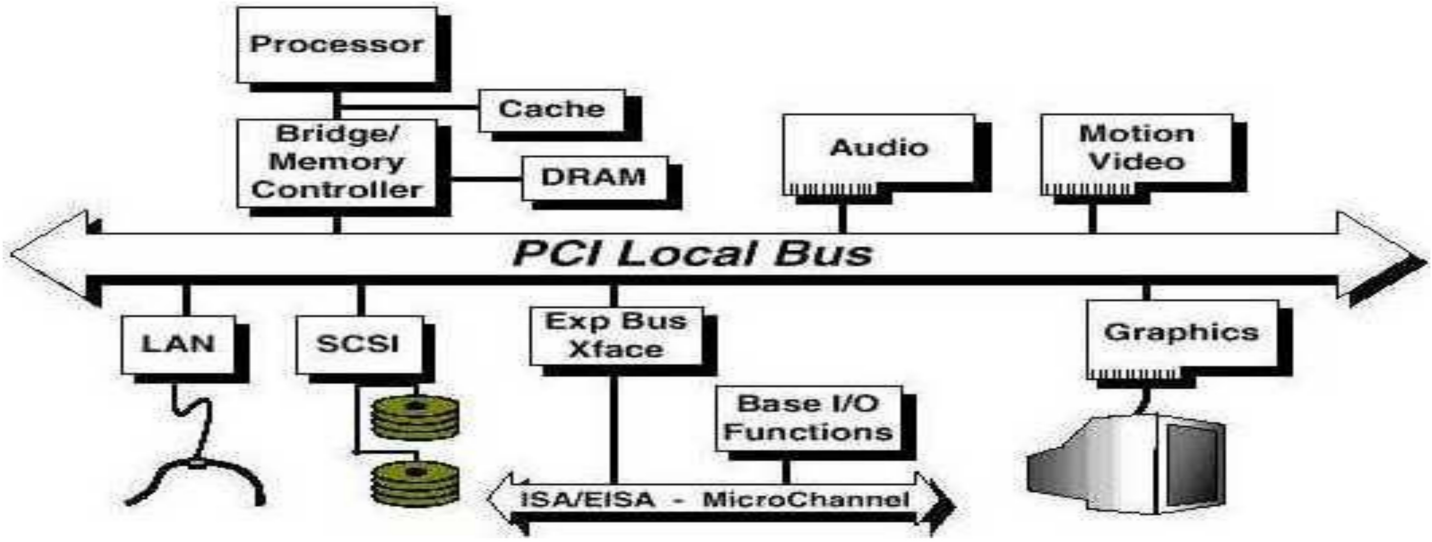


PCI Bus



- PCI is a 32-bit computer bus that is also available as a 64-bit bus.
- Data and address lines are multiplexed
- PCI bus allows a peripheral connected on this bus to take control and transfer data directly towards another peripheral PCI, this is called the DMA

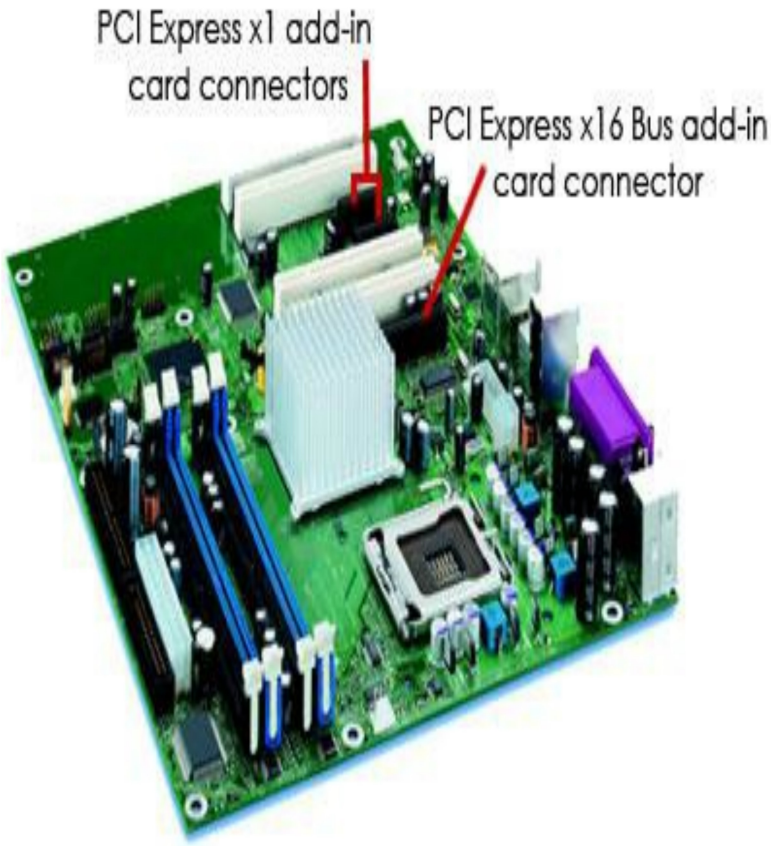
Peripheral Component Interconnect Bus (PCI)



- PCI cards :
  - network cards, sound cards, modems, extra ports such as USB or serial, TV tuner cards and disk controllers.
  - Historically video cards were typically PCI devices, but growing bandwidth requirements soon outgrew the capabilities of PCI.
  - The PCI Local Bus remains a very common interface in modern PCs, despite the availability of faster interfaces such as PCI-X and PCI Express

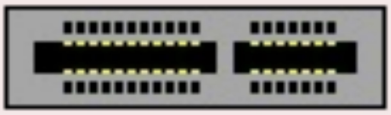
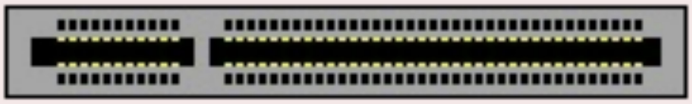
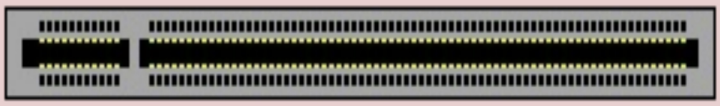
PCI Express Bus

- PCIe has
  - Higher throughput
  - lower I/O pin count
  - Smaller physical footprint
  - Better performance-scaling
  - More detailed error detection and reporting mechanism
- PCI Express bus is hot pluggable



PCI Express Bus

- The PCI Express bus comes in several versions

1X connector has 36 pins and is intended for high-bandwidth I/O use	
4X connector has 64 pins and is intended to be used on servers	
8X connector has 98 pins and is intended to be used on servers	
16X connector has 164 pins, is 89 mm long and is intended to be used on the graphics port	

Universal Serial Bus (USB)

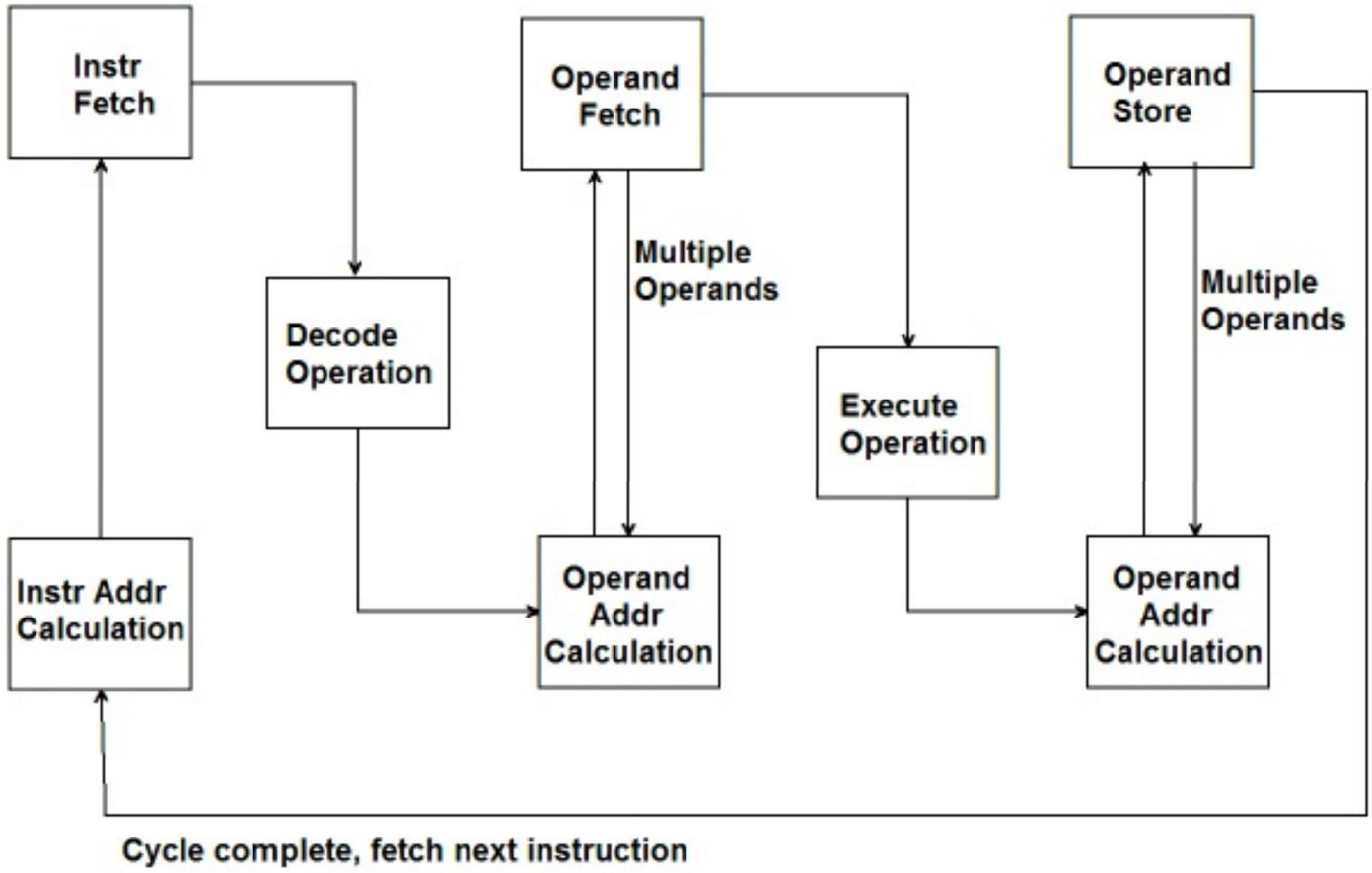
- USB has the capability of transferring 12 Mbps, supporting up to 127 devices.
- USB had three versions:
  - **USB 1.0** : The original release of USB supports 127 devices transferring 12 Mbps.
  - **USB 1.1**
    - known as full-speed USB,
    - supports a rate of 12 Mbps.
  - **USB 2.0**
    - known as hi-speed USB.
    - supporting a transfer rate of up to 480 Mbps
    - capable of supporting USB 1.0 and 1.1 devices and cables.

Instruction Execution Cycle

- Computers main job is to run programs.
- A program is simply a list of unambiguous instructions meant to be followed mechanically by a computer.
- A computer is built to carry out instructions that are written in machine language.
- A computer to execute a program, the program must be resident in RAM

Instruction Execution Cycle

- CPU is executing programs using the fetch-execute cycle
- the fetch-execute cycle consists of five steps :
  - Instruction Fetch (IF)
  - Instruction Decode (ID)
  - Data Fetch (DF) / Operand Fetch (OF)
  - Instruction Execution (EX)
  - Result Return (RR) / Store (ST)


Instr Fetch
Operand Fetch
Operand Store
Multiple Operands
Multiple Operands
Decode Operation
Execute Operation
Instr Addr Calculation
Operand Addr Calculation
Operand Addr Calculation
Cycle complete, fetch next instruction

Fetch-execute Cycle

- We can summarize the fetch –execution cycle in the next few steps explaining each register role as :
  - instruction fetch
    - MAR ← PC
    - send read signal to memory
    - MBR ←data from memory
    - IR ← MBR
  - decode operation: circuits in control unit analyze instruction to determine the operation and what operands are needed
  - calculate address of next instruction: PC ← PC + (size of current instruction)

Fetch-execute Cycle

- calculate address of operand: if operand requires memory access or I/O, determine the address of the operand
- fetch operand
  - MAR ← address of operand OR I/O AR ← address of operand
  - send read signal to memory or input device
  - MBR ←data from memory --OR-- I/O BR ← data from input device
- execute (arithmetic or logic operation)
- store operand
  - MAR ← address to store operand OR I/O AR← address to store operand
  - MBR ← --OR-- I/O BR ← data from input device
  - send write signal to memory or output device