Chapter – 1
Introduction

1.1 Computer Organization and Architecture

- Computer Architecture refers to those attributes of a system that have a direct impact on the logical execution of a program. Examples:
  - the instruction set
  - the number of bits used to represent various data types
  - I/O mechanisms
  - memory addressing techniques
- Computer Organization refers to the operational units and their interconnections that realize the architectural specifications. Examples are things that are transparent to the programmer:
  - control signals
  - interfaces between computer and peripherals
  - the memory technology being used
- So, for example, the fact that a multiply instruction is available is a computer architecture issue. How that multiply is implemented is a computer organization issue.

- Architecture is those attributes visible to the programmer
  - Instruction set, number of bits used for data representation, I/O mechanisms, addressing techniques.
  - e.g. Is there a multiply instruction?
- Organization is how features are implemented
  - Control signals, interfaces, memory technology.
  - e.g. Is there a hardware multiply unit or is it done by repeated addition?
- All Intel x86 family share the same basic architecture
- The IBM System/370 family share the same basic architecture
- This gives code compatibility
  - At least backwards
- Organization differs between different versions

1.2 Structure and Function

- Structure is the way in which components relate to each other
- Function is the operation of individual components as part of the structure
- All computer functions are:
  - **Data processing**: Computer must be able to process data which may take a wide variety of forms and the range of processing.
  - **Data storage**: Computer stores data either temporarily or permanently.
  - **Data movement**: Computer must be able to move data between itself and the outside world.
  - **Control**: There must be a control of the above three functions.

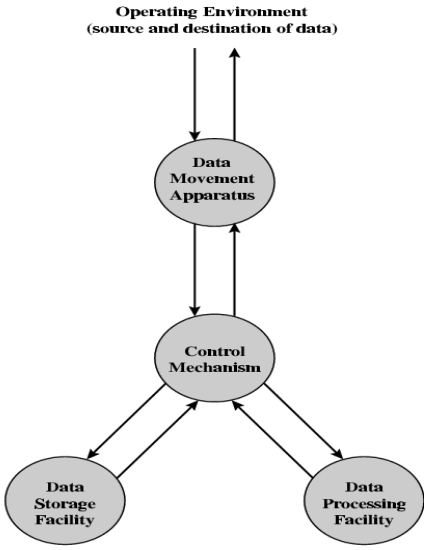


Fig: Functional view of a computer

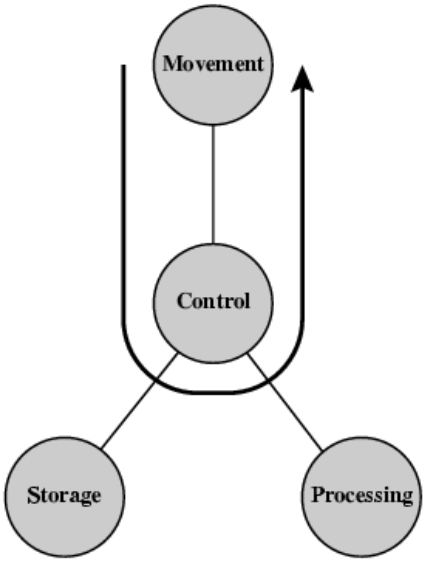


Fig: Data movement operation

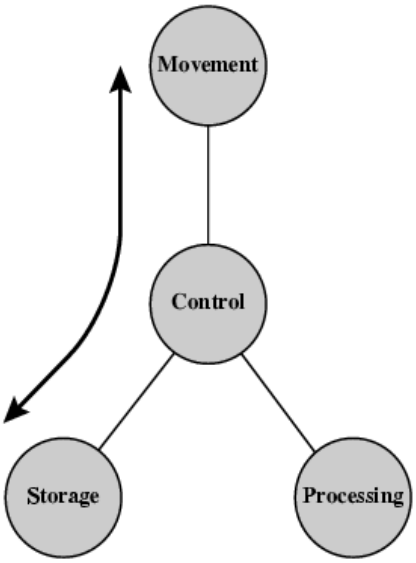


Fig: Storage Operation

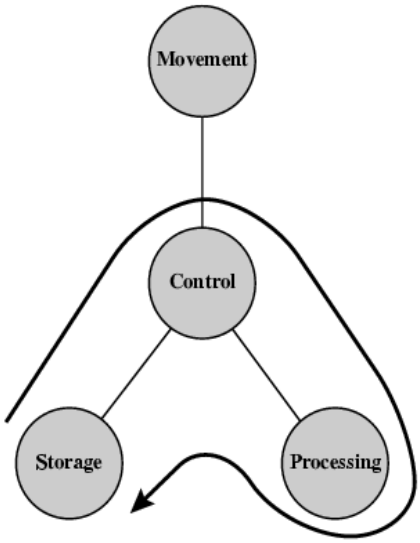


Fig: Processing from / to storage

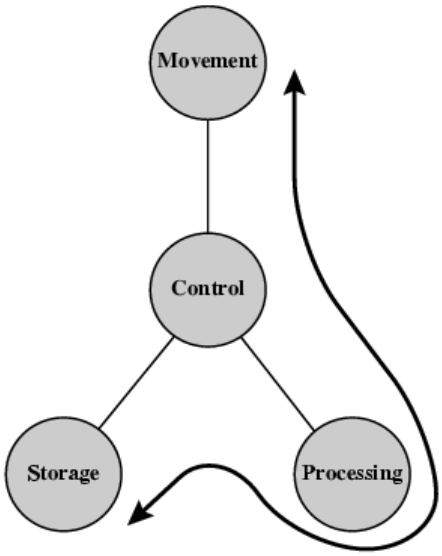


Fig: Processing from storage to i/o

- Four main structural components:
  - Central processing unit (CPU)
  - Main memory
  - I / O
  - System interconnections
- CPU structural components:
  - Control unit
  - Arithmetic and logic unit (ALU)
  - Registers
  - CPU interconnections

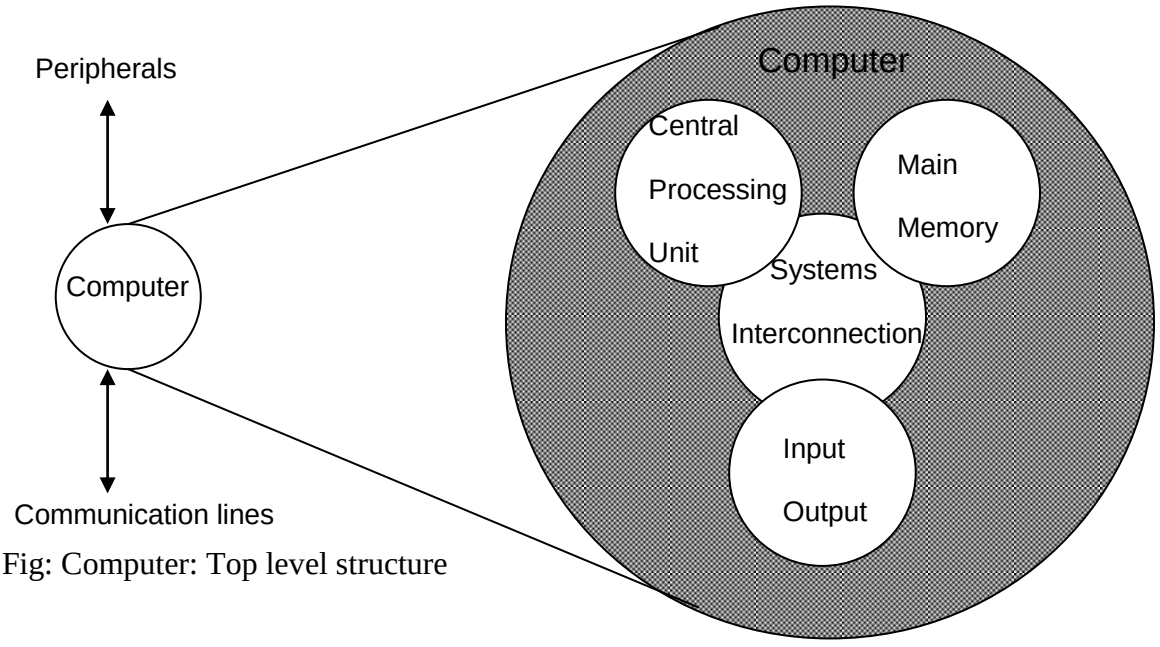


Fig: Computer: Top level structure

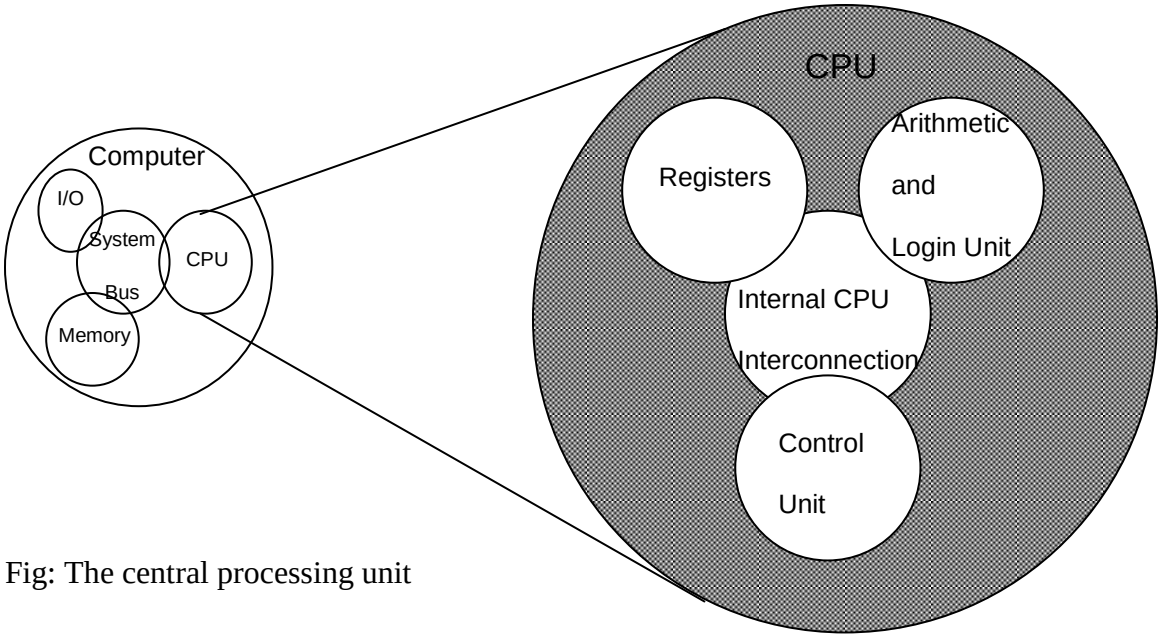


Fig: The central processing unit

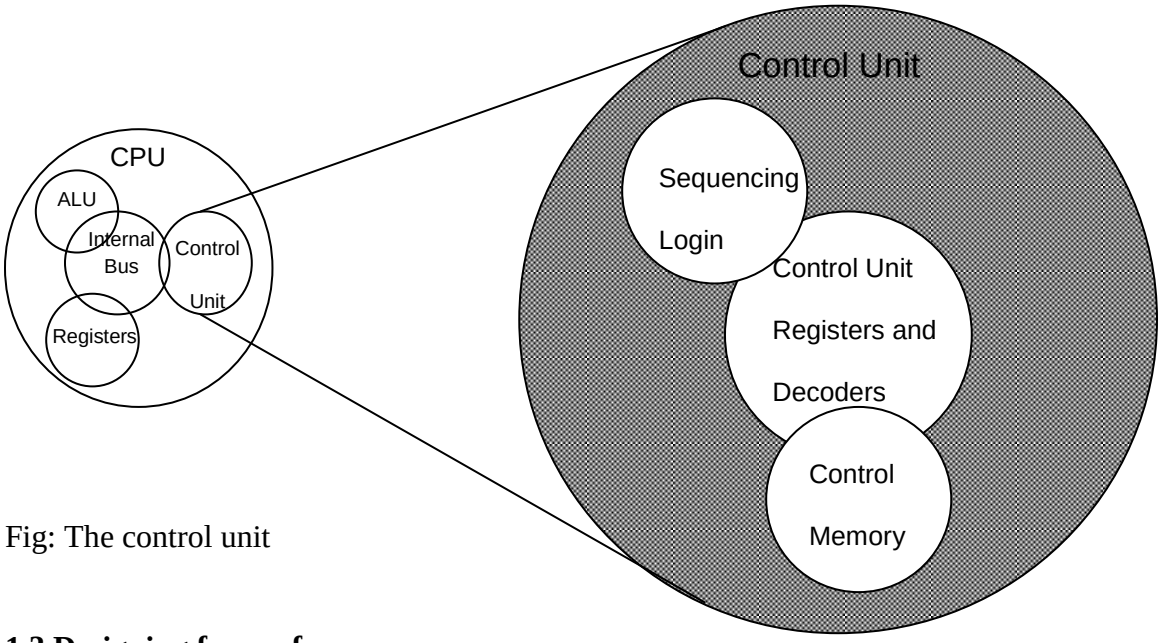


Fig: The control unit

1.3 Designing for performance

Some of the driving factors behind the need to design for performance:

- Microprocessor Speed
  - Pipelining
  - On board cache, on board L1 & L2 cache
  - Branch prediction: The processor looks ahead in the instruction code fetched from memory and predicts which branches, or group of instructions are likely to be processed next.
  - Data flow analysis: The processor analyzes which instructions are dependent on each other's results, or data, to create an optimized schedule of instructions to prevent delay.

- Speculative execution: Using branch prediction and data flow analysis, some processors speculatively execute instructions ahead of their actual appearance in the program execution, holding the results in temporary locations.

- Performance Mismatch
  - Processor speed increased
  - Memory capacity increased
  - Memory speed lags behind processor speed

Below figure depicts the history; while processor speed and memory capacity have grown rapidly, the speed with which data can be transferred between main memory and the processor has lagged badly.

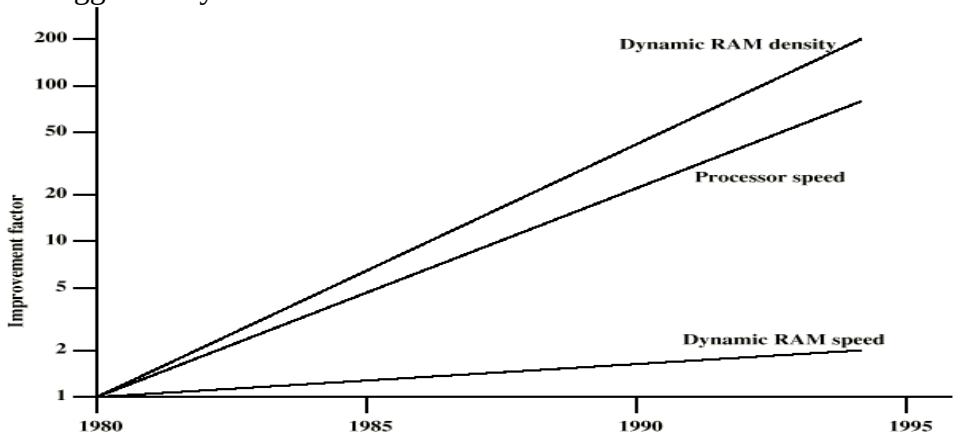


Fig: Evolution of DRAM and processor Characteristics

The effects of these trends are shown vividly in figure below. The amount of main memory needed is going up, but DRAM density is going up faster (number of DRAM per system is going down).

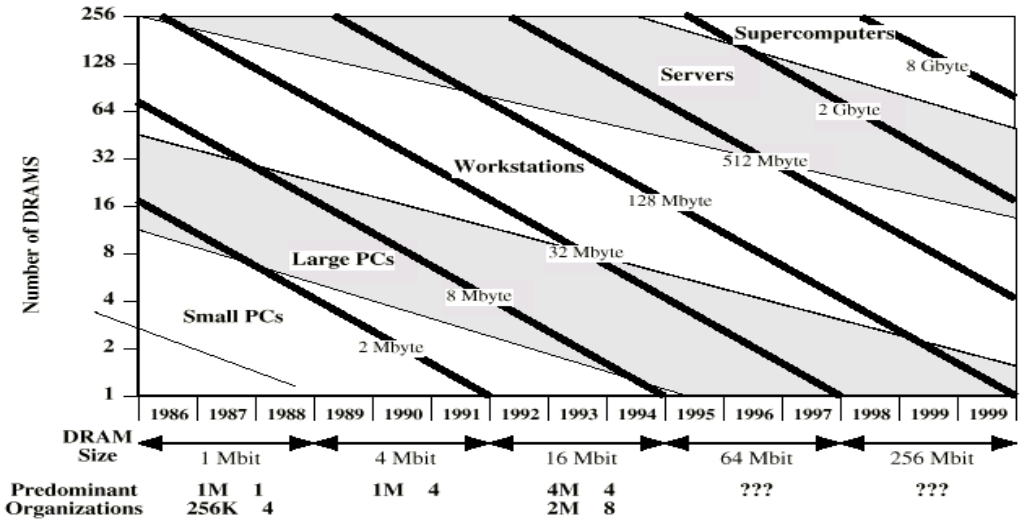


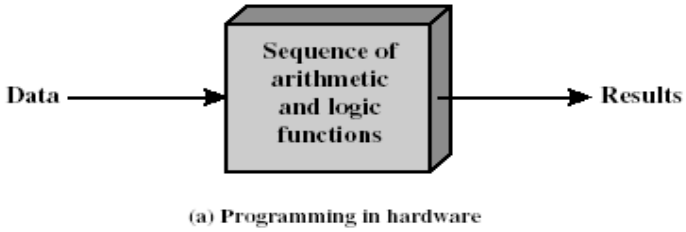
Fig: Trends in DRAM use

Solutions

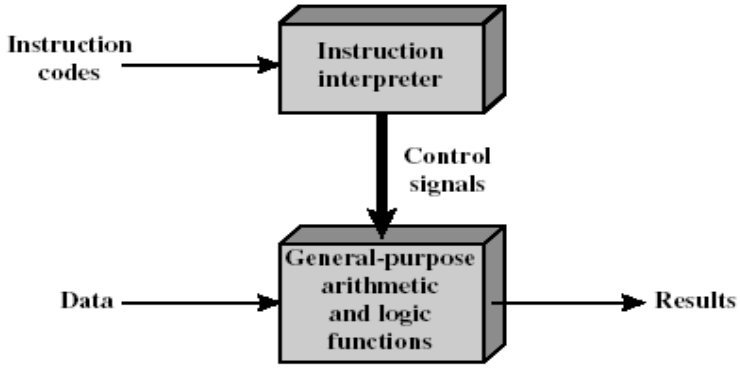
- Increase number of bits retrieved at one time
  - Make DRAM "wider" rather than "deeper" to use wide bus data paths.
- Change DRAM interface
  - Cache
- Reduce frequency of memory access
  - More complex cache and cache on chip
- Increase interconnection bandwidth
  - High speed buses
  - Hierarchy of buses

1.4 Computer Components

- The Control Unit (CU) and the Arithmetic and Logic Unit (ALU) constitute the Central Processing Unit (CPU)
- Data and instructions need to get into the system and results need to get out
  - Input/output (I/O module)
- Temporary storage of code and results is needed
  - Main memory (RAM)
- Program Concept
  - Hardwired systems are inflexible
  - General purpose hardware can do different tasks, given correct control signals
  - Instead of re-wiring, supply a new set of control signals

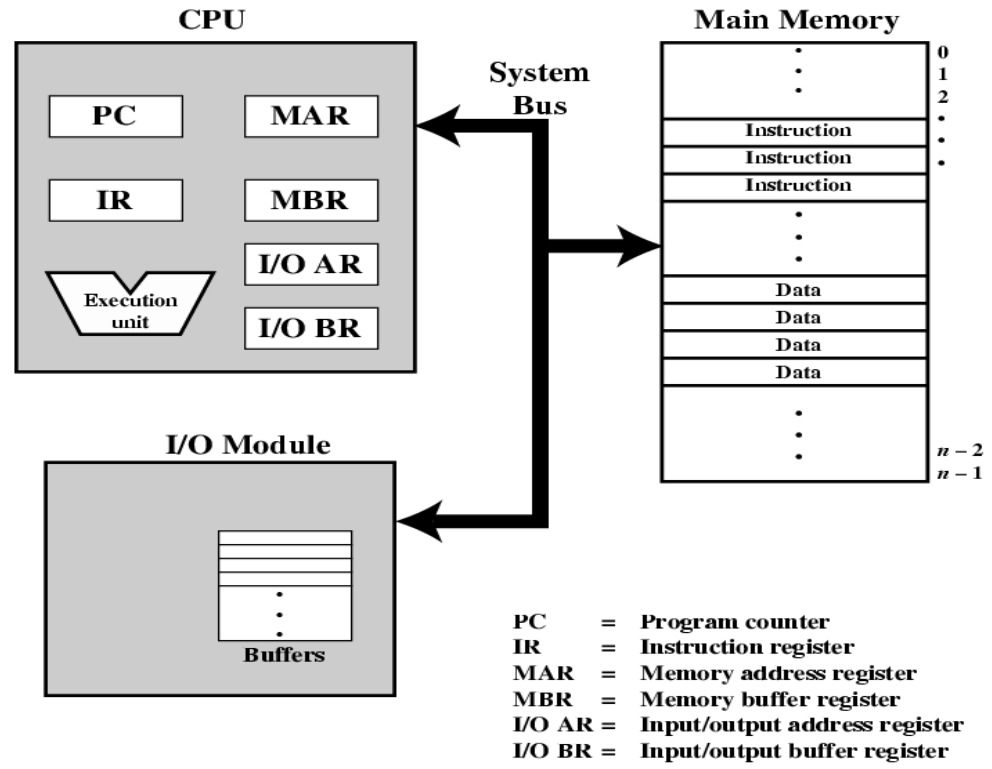


(a) Programming in hardware



(b) Programming in software

Fig: Hardware and Software Approaches



1.5 Computer Function

The basic function performed by a computer is execution of a program, which consists of a set of instructions stored in memory.

- Two steps of Instructions Cycle:
  - Fetch
  - Execute

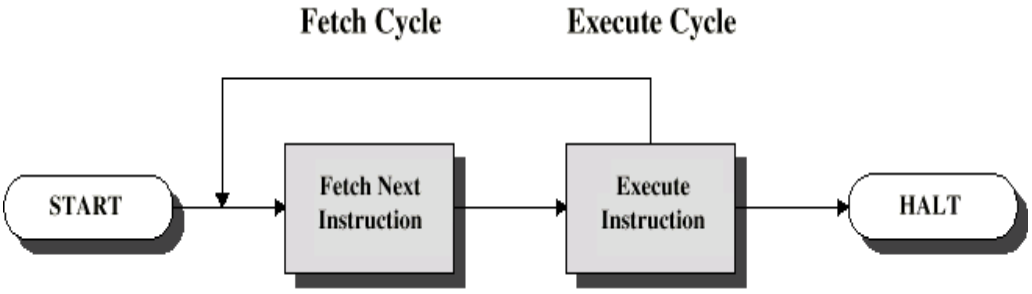


Fig: Basic Instruction Cycle

- Fetch Cycle
  - Program Counter (PC) holds address of next instruction to fetch
  - Processor fetches instruction from memory location pointed to by PC
  - Increment PC
    - Unless told otherwise
  - Instruction loaded into Instruction Register (IR)

- Execute Cycle
  - Processor interprets instruction and performs required actions, such as:
    - Processor - memory
      - data transfer between CPU and main memory
    - Processor - I/O
      - Data transfer between CPU and I/O module
    - Data processing
      - Some arithmetic or logical operation on data
    - Control
      - Alteration of sequence of operations
      - e.g. jump
    - Combination of above

Example of program execution.

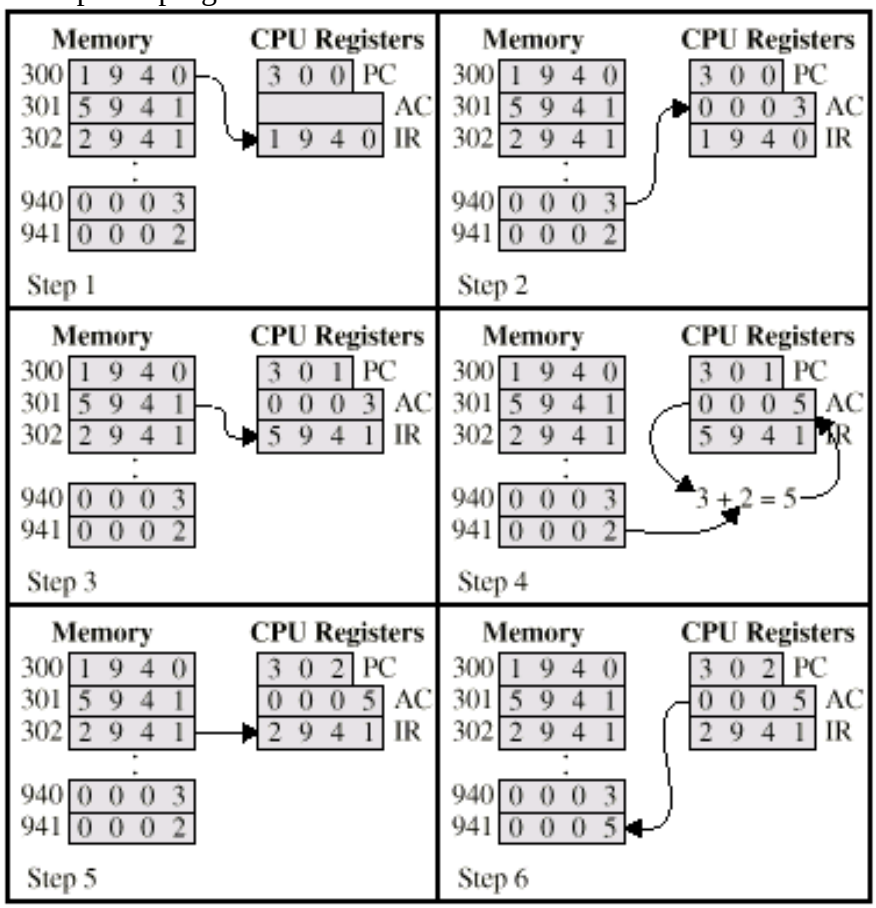


Fig: Example of program execution (consists of memory and registers in hexadecimal)

- The PC contains 300, the address of the first instruction. The instruction (the value 1940 in hex) is loaded into IR and PC is incremented. This process involves the use of MAR and MBR.
- The first hexadecimal digit in IR indicates that the AC is to be loaded. The remaining three hexadecimal digits specify the address (940) from which data are to be loaded.
- The next instruction (5941) is fetched from location 301 and PC is incremented.

- The old contents of AC and the contents of location 941 are added and the result is stored in the AC.
- The next instruction (2941) is fetched from location 302 and the PC is incremented.
- The contents of the AC are stored in location 941.

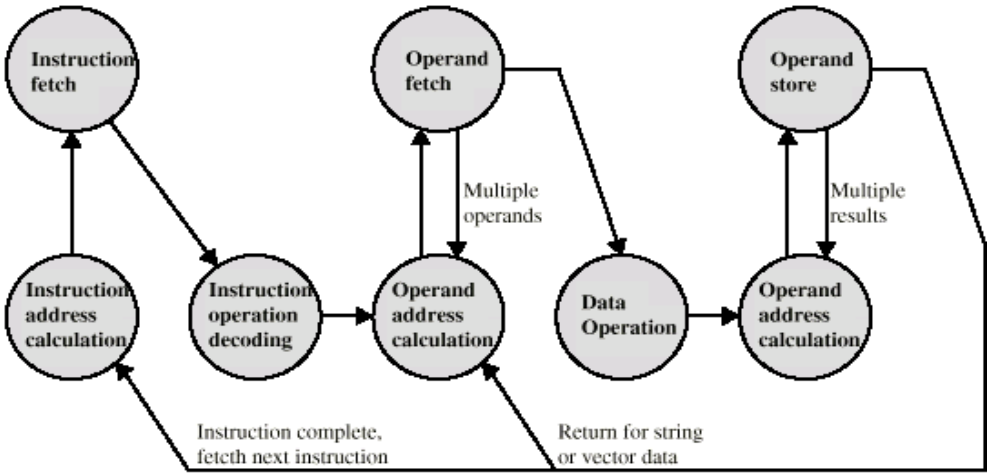
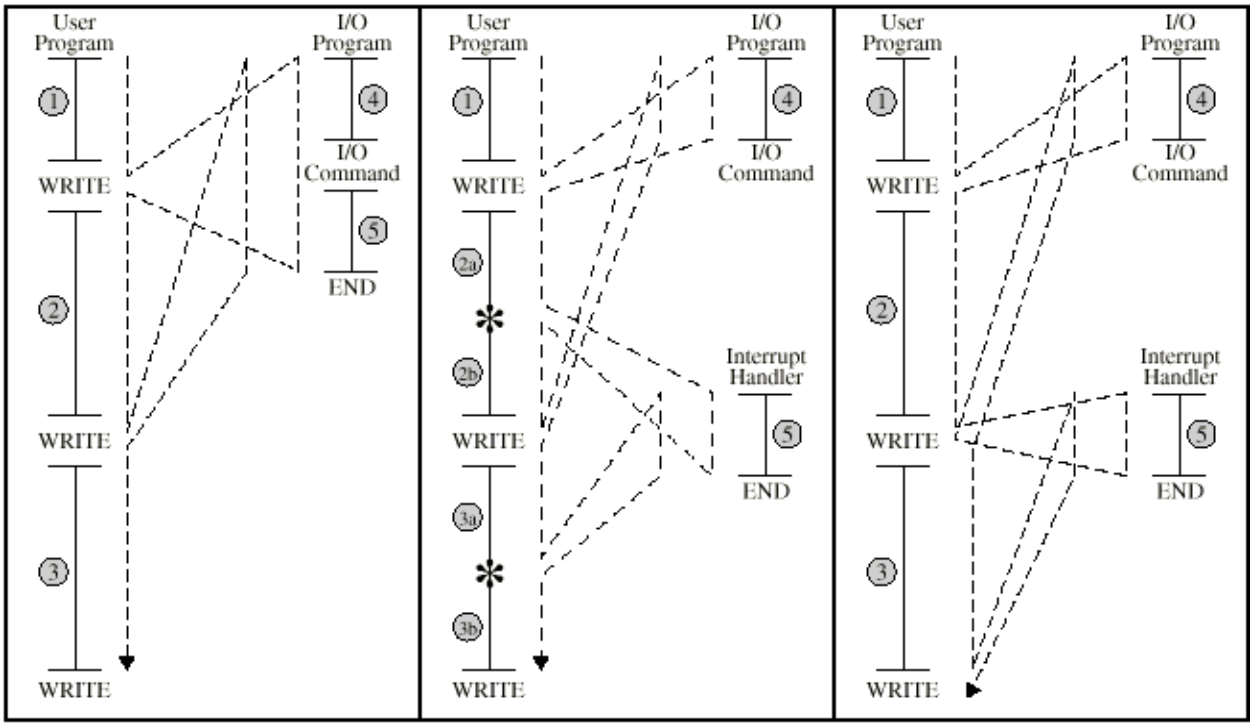


Fig: Instruction cycle state diagram

Interrupts:

- Mechanism by which other modules (e.g. I/O) may interrupt normal sequence of processing
- Program
  - e.g. overflow, division by zero
- Timer
  - Generated by internal processor timer
  - Used in pre-emptive multi-tasking
- I/O
  - from I/O controller
- Hardware failure
  - e.g. memory parity error



- Indicated by an interrupt signal
- If no interrupt, fetch next instruction
- If interrupt pending:
  - Suspend execution of current program
  - Save context
  - Set PC to start address of interrupt handler routine
  - Process interrupt
  - Restore context and continue interrupted program

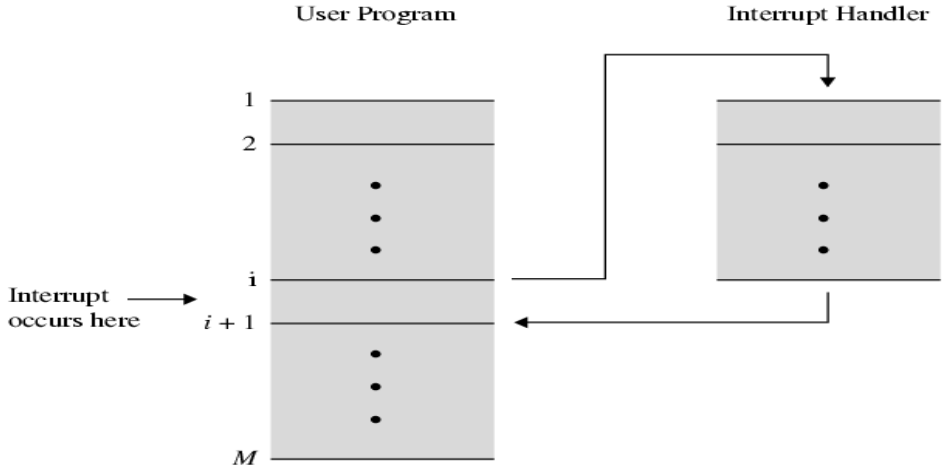


Fig: Transfer of control via interrupts

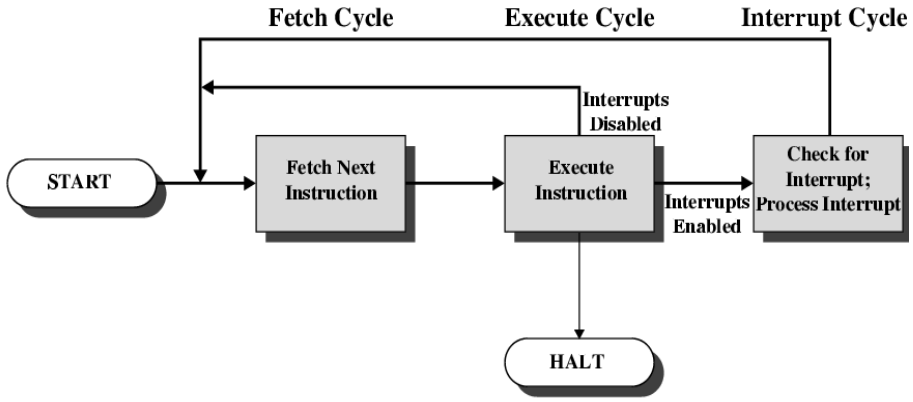


Fig: Instruction Cycle with Interrupts

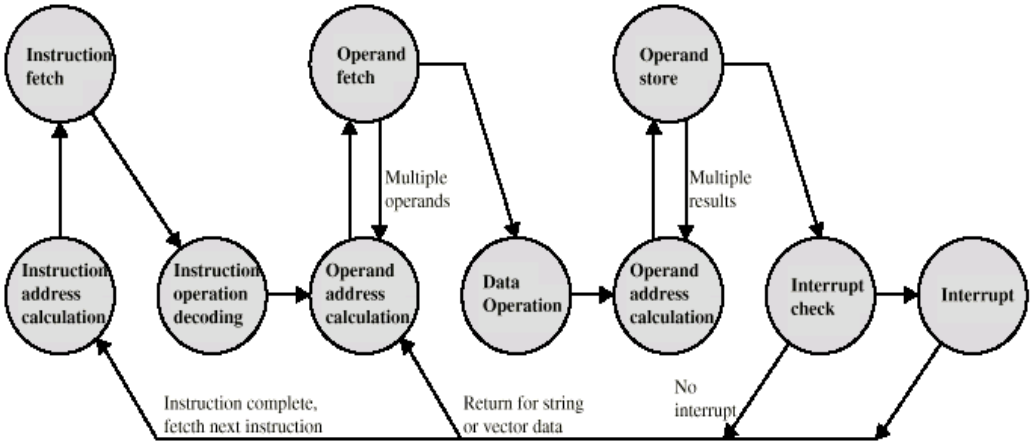


Fig: Instruction cycle state diagram, with interrupts

- Multiple Interrupts
  - Disable interrupts (approach #1)
    - Processor will ignore further interrupts whilst processing one interrupt
    - Interrupts remain pending and are checked after first interrupt has been processed
    - Interrupts handled in sequence as they occur
  - Define priorities (approach #2)
    - Low priority interrupts can be interrupted by higher priority interrupts
    - When higher priority interrupt has been processed, processor returns to previous interrupt

1.6 Interconnection structures

The collection of paths connecting the various modules is called the interconnecting structure.

- All the units must be connected
- Different type of connection for different type of unit
  - Memory
  - Input/Output
  - CPU

- Memory Connection
  - Receives and sends data
  - Receives addresses (of locations)
  - Receives control signals
    - Read
    - Write
    - Timing

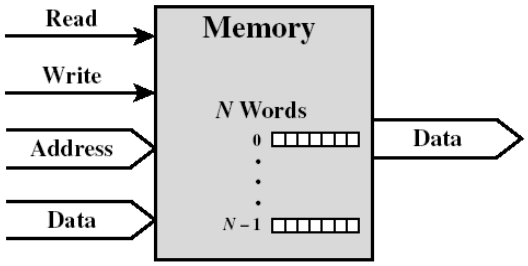


Fig: Memory Module

- I/O Connection
  - Similar to memory from computer's viewpoint
  - Output
    - Receive data from computer
    - Send data to peripheral
  - Input
    - Receive data from peripheral
    - Send data to computer
  - Receive control signals from computer
  - Send control signals to peripherals
    - e.g. spin disk
  - Receive addresses from computer
    - e.g. port number to identify peripheral
  - Send interrupt signals (control)

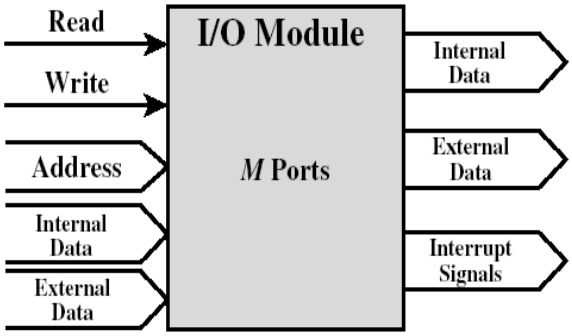


Fig: I/O Module

- CPU Connection
  - Reads instruction and data
  - Writes out data (after processing)
  - Sends control signals to other units
  - Receives (& acts on) interrupts

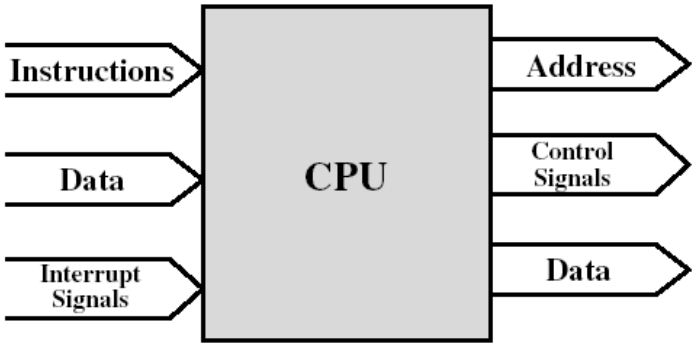


Fig: CPU Module

1.7 Bus interconnection

- A bus is a communication pathway connecting two or more devices
- Usually broadcast (all components see signal)
- Often grouped
  - A number of channels in one bus
  - e.g. 32 bit data bus is 32 separate single bit channels
- Power lines may not be shown
- There are a number of possible interconnection systems
- Single and multiple BUS structures are most common
- e.g. Control/Address/Data bus (PC)
- e.g. Unibus (DEC-PDP)
- Lots of devices on one bus leads to:
  - Propagation delays
  - Long data paths mean that co-ordination of bus use can adversely affect performance
  - If aggregate data transfer approaches bus capacity
- Most systems use multiple buses to overcome these problems

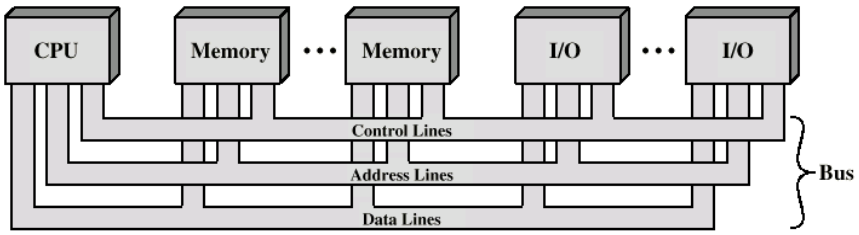


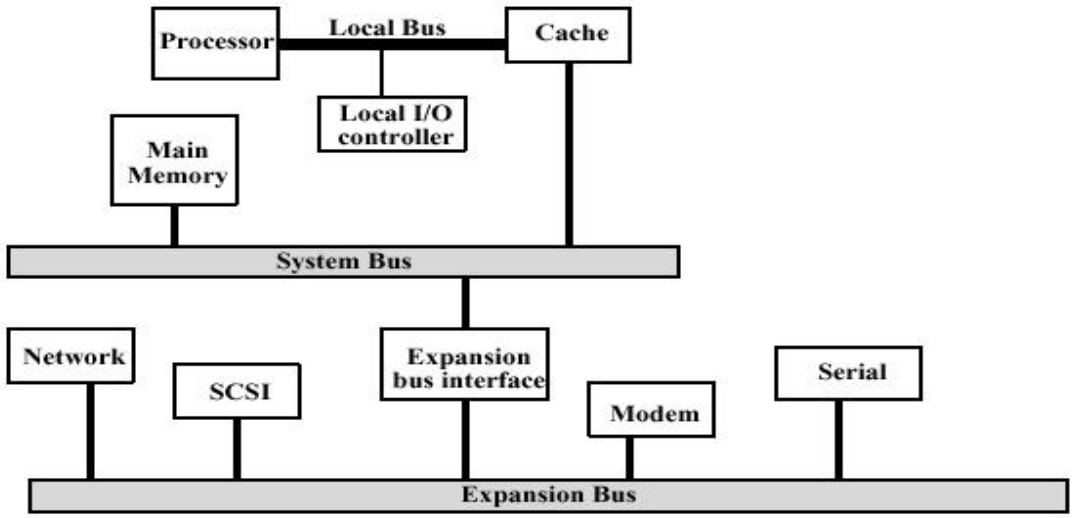
Fig: Bus Interconnection Scheme

- Data Bus
  - Carries data
    - Remember that there is no difference between "data" and "instruction" at this level
  - Width is a key determinant of performance
    - 8, 16, 32, 64 bit
- Address Bus
  - Identify the source or destination of data
  - e.g. CPU needs to read an instruction (data) from a given location in memory
  - Bus width determines maximum memory capacity of system
    - e.g. 8080 has 16 bit address bus giving 64k address space
- Control Bus
  - Control and timing information
    - Memory read
    - Memory write
    - I/O read
    - I/O write
    - Transfer ACK
    - Bus request
    - Bus grant
    - Interrupt request
    - Interrupt ACK
    - Clock
    - Reset

Multiple Bus Hierarchies

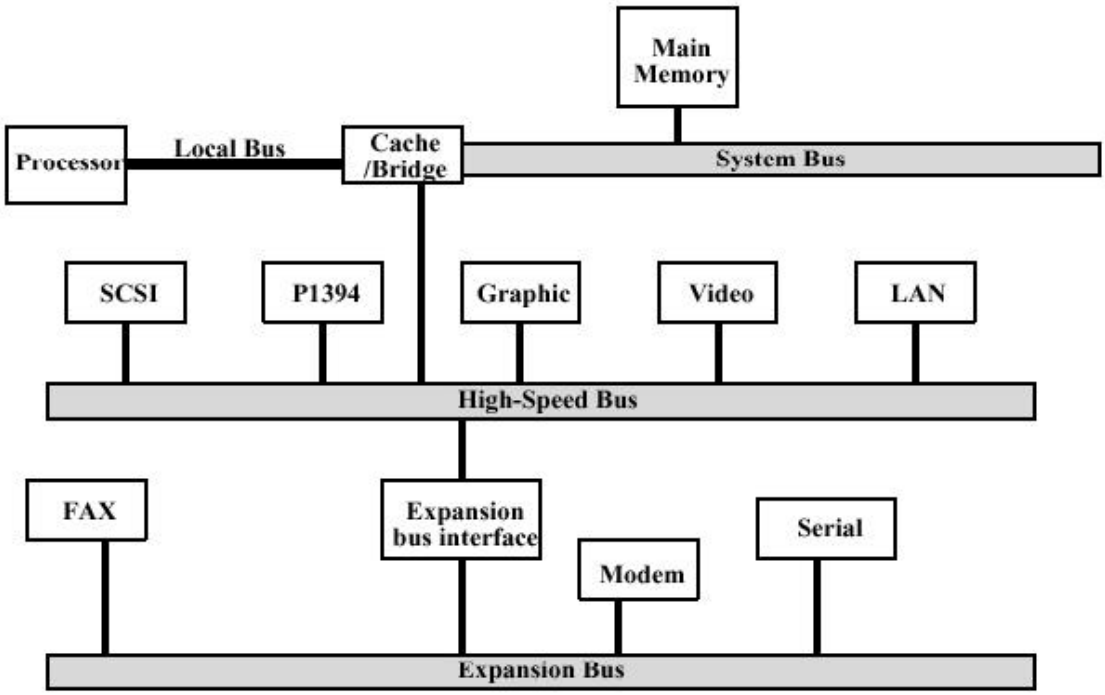
- A great number of devices on a bus will cause performance to suffer
  - Propagation delay - the time it takes for devices to coordinate the use of the bus
  - The bus may become a bottleneck as the aggregate data transfer demand approaches the capacity of the bus (in available transfer cycles/second)
- Traditional Hierarchical Bus Architecture
  - Use of a cache structure insulates CPU from frequent accesses to main memory
  - Main memory can be moved off local bus to a system bus
  - Expansion bus interface
    - buffers data transfers between system bus and I/O controllers on expansion bus
    - insulates memory-to-processor traffic from I/O traffic

Traditional Hierarchical Bus Architecture Example



- High-performance Hierarchical Bus Architecture
  - Traditional hierarchical bus breaks down as higher and higher performance is seen in the I/O devices
  - Incorporates a high-speed bus
    - specifically designed to support high-capacity I/O devices
    - brings high-demand devices into closer integration with the processor and at the same time is independent of the processor
    - Changes in processor architecture do not affect the high-speed bus, and vice versa
  - Sometimes known as a mezzanine architecture

High-performance Hierarchical Bus Architecture Example



Elements of Bus Design

- Bus Types
  - Dedicated
    - Separate data & address lines
  - Multiplexed
    - Shared lines
    - Address valid or data valid control line
    - Advantage - fewer lines
    - Disadvantages
      - More complex control
      - Ultimate performance
- Bus Arbitration
  - More than one module controlling the bus
    - e.g. CPU and DMA controller
  - Only one module may control bus at one time
  - Arbitration may be centralised or distributed
- Centralised Arbitration
  - Single hardware device controlling bus access
    - Bus Controller
    - Arbiter
  - May be part of CPU or separate
- Distributed Arbitration

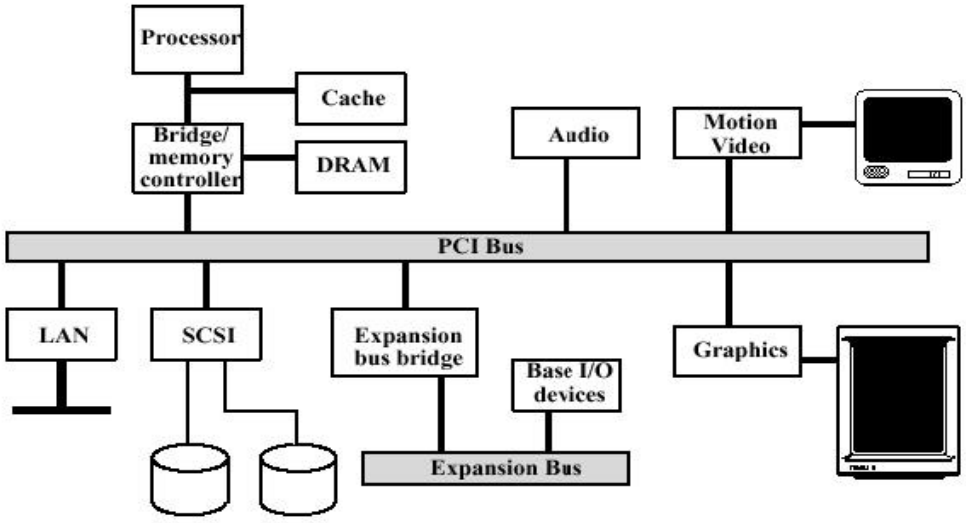
- Each module may claim the bus
- Control logic on all modules

- Timing
  - Co-ordination of events on bus
  - Synchronous
    - Events determined by clock signals
    - Control Bus includes clock line
    - A single 1-0 is a bus cycle
    - All devices can read clock line
    - Usually sync on leading edge
    - Usually a single cycle for an event
- Bus Width
  - Address: Width of address bus has an impact on system capacity i.e. wider bus means greater the range of locations that can be transferred.
  - Data: width of data bus has an impact on system performance i.e. wider bus means number of bits transferred at one time.
- Data Transfer Type
  - Read
  - Write
  - Read-modify-write
  - Read-after-write
  - Block

1.8 PCI

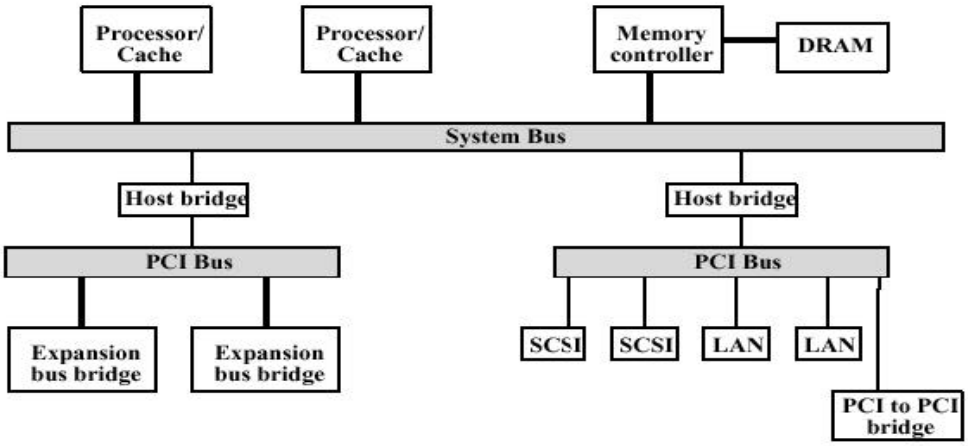
- PCI is a popular high bandwidth, processor independent bus that can function as mezzanine or peripheral bus.
- PCI delivers better system performance for high speed I/O subsystems (graphic display adapters, network interface controllers, disk controllers etc.)
- PCI is designed to support a variety of microprocessor based configurations including both single and multiple processor system.
- It makes use of synchronous timing and centralised arbitration scheme.
- PCI may be configured as a 32 or 64-bit bus.
- Current Standard
  - up to 64 data lines at 33Mhz
  - requires few chips to implement
  - supports other buses attached to PCI bus
  - public domain, initially developed by Intel to support Pentium-based systems
  - supports a variety of microprocessor-based configurations, including multiple processors
  - uses synchronous timing and centralized arbitration

Typical Desktop System



Note: Bridge acts as a data buffer so that the speed of the PCI bus may differ from that of the processor's I/O capability.

Typical Server System



Note: In a multiprocessor system, one or more PCI configurations may be connected by bridges to the processor's system bus.

PCI Bus Lines

- Systems lines
  - Including clock and reset
- Address & Data
  - 32 time mux lines for address/data
  - Interrupt & validate lines
- Interface Control
- Arbitration
  - Not shared
  - Direct connection to PCI bus arbiter

- Error lines
- Interrupt lines
    - Not shared
- Cache support
- 64-bit Bus Extension
    - Additional 32 lines
    - Time multiplexed
    - 2 lines to enable devices to agree to use 64-bit transfer
- JTAG/Boundary Scan
    - For testing procedures

PCI Commands

- Transaction between initiator (master) and target
- Master claims bus
- Determine type of transaction
    - e.g. I/O read/write
- Address phase
- One or more data phases

PCI Enhancements: AGP

- AGP – Advanced Graphics Port
    - Called a port, not a bus because it only connects 2 devices