

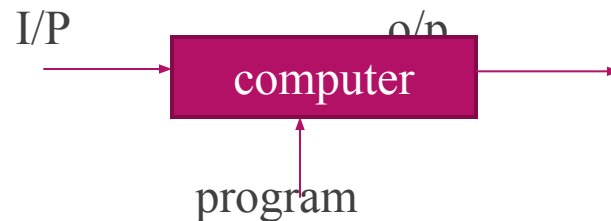


MODULE-1

Introduction to Computer Organization

What is a Computer organisation?

- ▶ Describes function & design of various units of digital computers
- ▶ Deals with Computer Hardware and Architecture
- ▶ Computer/Digital Computer



- ▶ Classification of Computers

Range of powerful computers



Basic functional units



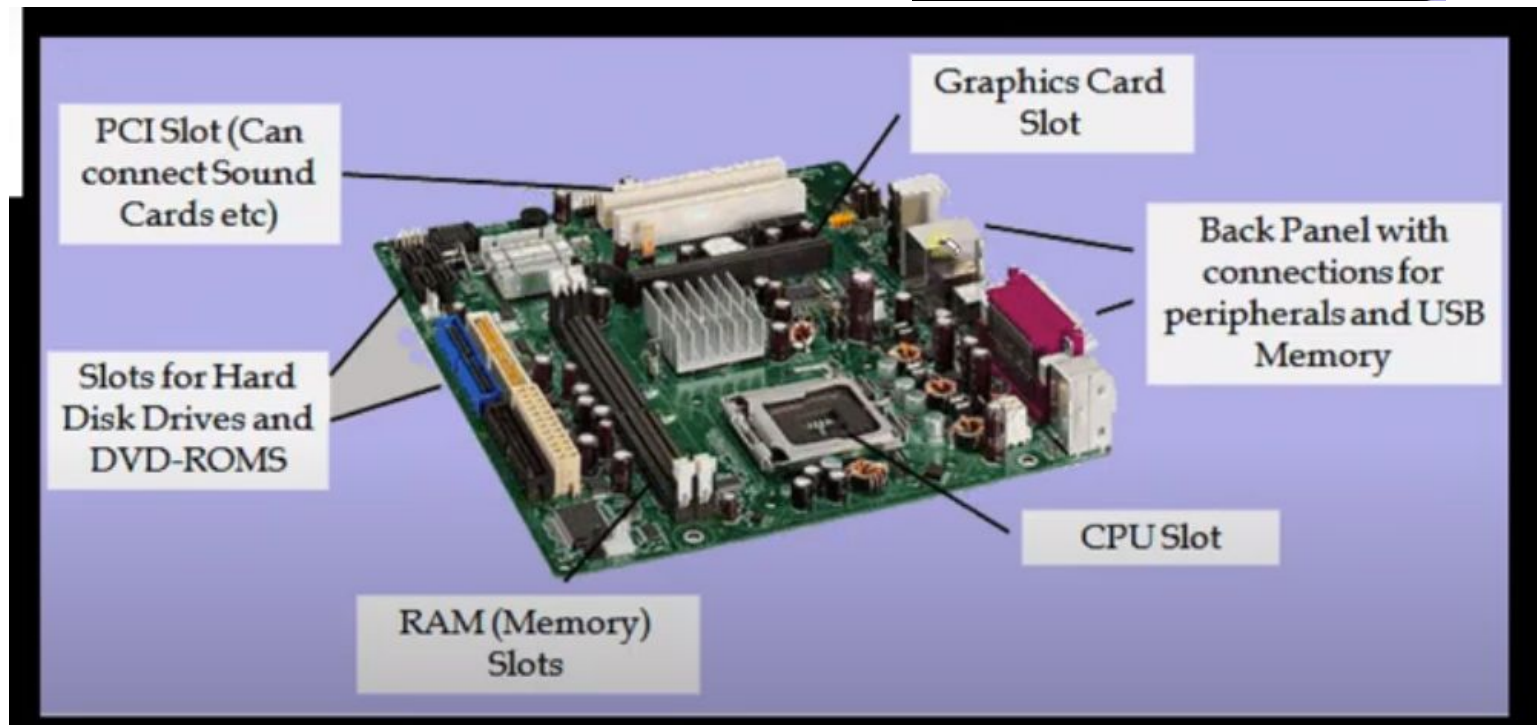
Continued...

- ▶ Input Unit-accepts information through input units, which read the data
- ▶ Memory Unit- 2 types
- ▶ ALU-Faster than other devices connected to system.
- ▶ Output unit-Counter part of input unit
- ▶ Control Unit-Sends control signal to other units and sense their states

Information Handled by a Computer

- ▶ Instructions/machine instructions
 - Govern the transfer of information within a computer as well as between the computer and its I/O devices
 - Specify the arithmetic and logic operations to be performed
 - Program
- ▶ Data
 - Used as operands by the instructions
 - Source program
- ▶ Encoded in binary code – 0 and 1

Motherboard



Input Devices



Output Devices



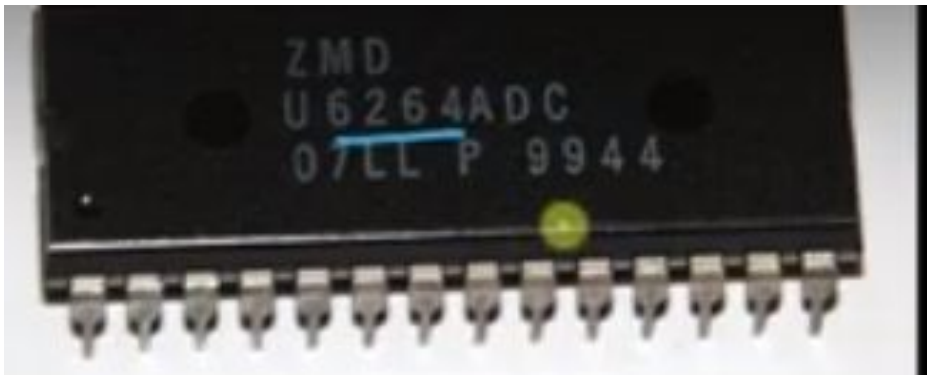
Memory Devices



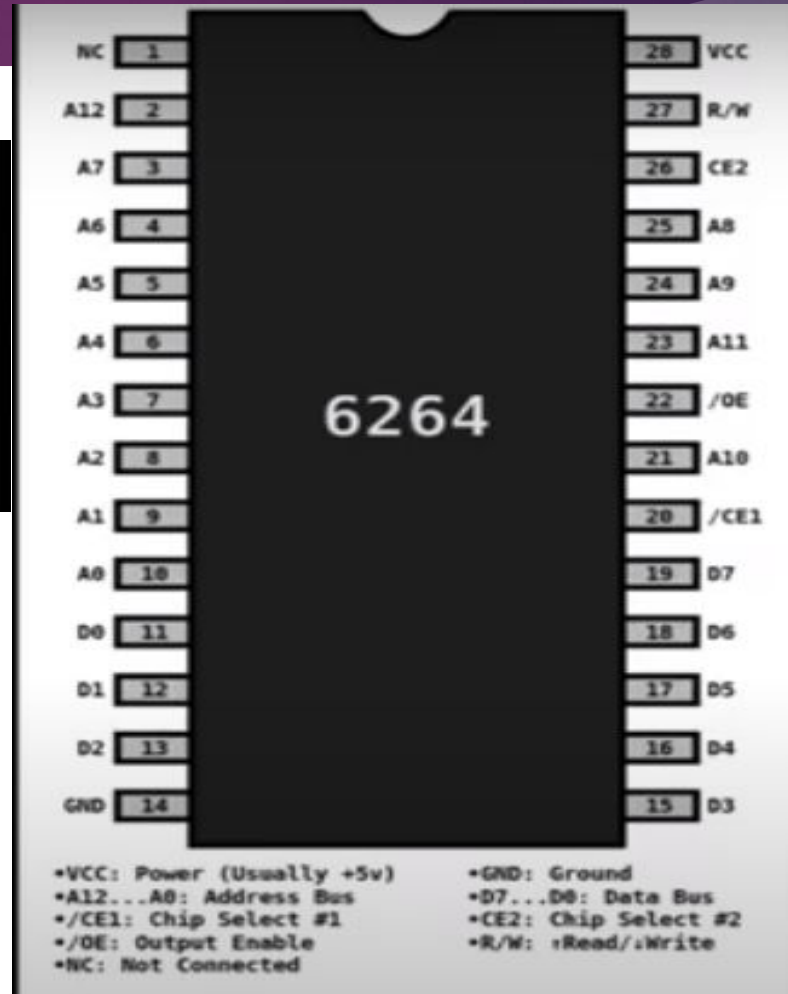
Memory Unit

- ▶ Store programs and data
- ▶ Two classes of storage
 - Primary storage
 - ❖ Fast
 - ❖ Programs must be stored in memory while they are being executed
 - ❖ Large number of semiconductor storage cells
 - ❖ Processed in words.
 - ❖ Address
 - ❖ Expensive.
 - Secondary storage – larger amount of data and cheaper

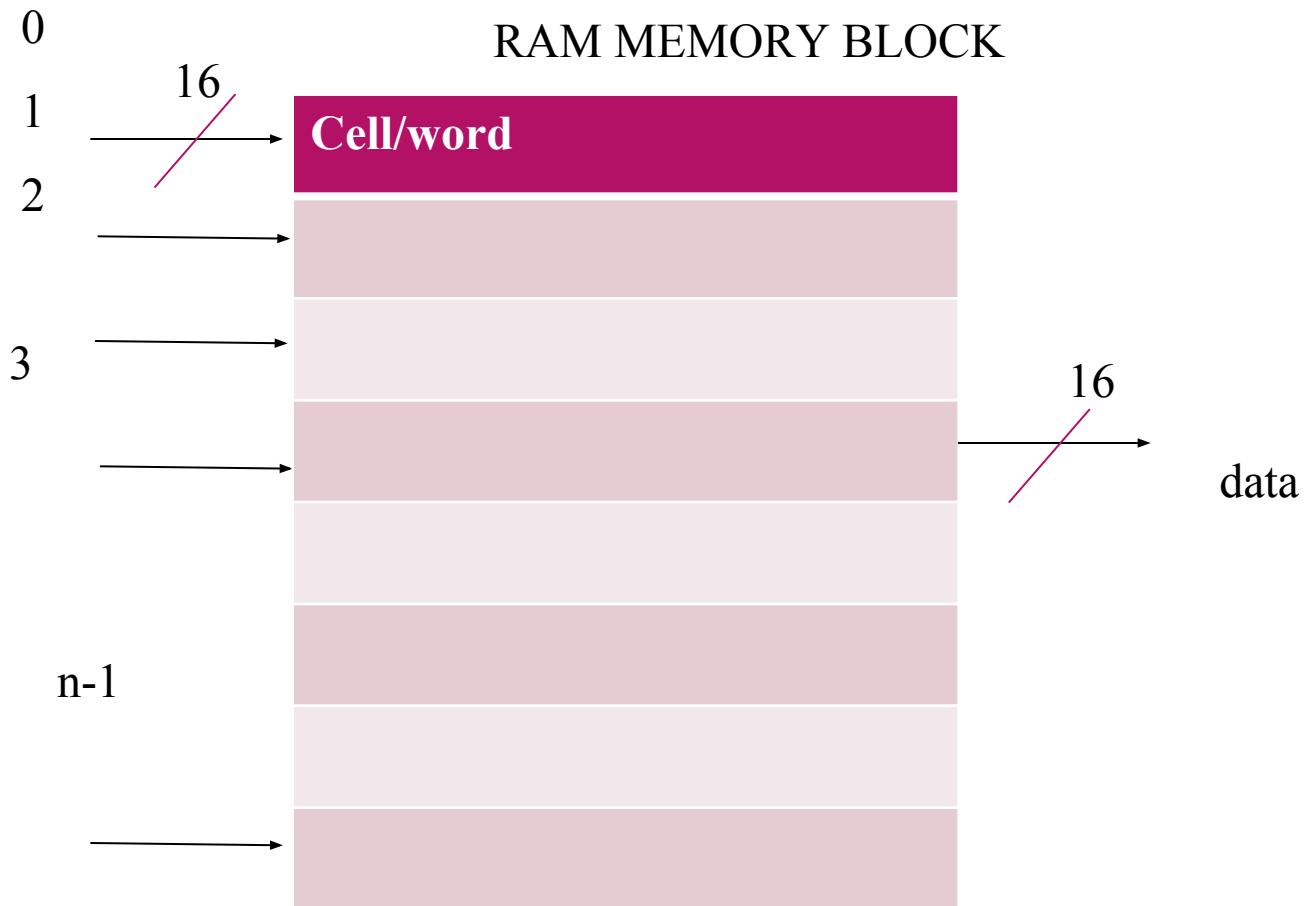
How memory chip looks like?



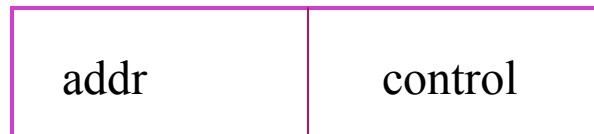
$$2^{12} \times 8$$
$$= 4K \times 8.R/W$$



RAM MEMORY BLOCK



- ▶ CPU wants data from cell 4(Read/write)
- ▶ CPU communicates R,W control signals along with address-memory requests



Address/Data

- ▶ Datasheet of memory chip-64K*16 or 64K*32.
- ▶ 2^{16} locations=(64k), cell width 16 bits.
- ▶ Word or cell widths.
- ▶ Cell width-width of data bus.
- ▶ **$2^k = n$ locations, k -no of address lines**

Control signals

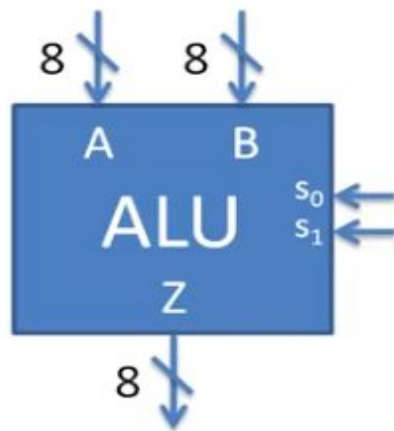
R/W

Arithmetic and Logic Unit (ALU)

- ▶ Most computer operations are executed in ALU of the processor.
- ▶ Load the operands into memory – bring them to the processor – perform operation in ALU – store the result back to memory or retain in the processor.
- ▶ Registers
- ▶ Fast control of ALU

Arithmetic and Logic Unit (ALU)

- ▶ Most computer operations are executed in ALU of the processor.



0 0 0 0 1 0 1 0 A

0 0 1 1 0 1 0 1 B

1 1 $s_1 s_0$

$s_1 s_0$	Operation
0 0	$Z=A+B$
0 1	$Z=A+1$
1 0	$Z=\text{NOT } A$

Control Unit


- ▶ All computer operations are controlled by the control unit.
- ▶ The timing signals that govern the I/O transfers are also generated by the control unit.
- ▶ Control unit is usually distributed throughout the machine instead of standing alone.
- ▶ Operations of a computer:
 - Accept information in the form of programs and data through an input unit and store it in the memory
 - Fetch the information stored in the memory, under program control, into an ALU, where the information is processed
 - Output the processed information through an output unit
 - Control all activities inside the machine through a control unit

Basic Operational Concepts

- ▶ To execute program, Sequence of instructions
- ▶ A Typical Instruction `ADD LOCA,R0`
- ▶ Add the operand at memory location LOCA to the operand in a register R0 in the processor.
- ▶ Place the sum into register R0.
- ▶ The original contents of LOCA are preserved.
- ▶ The original contents of R0 is overwritten.
- ▶ Instruction is fetched from the memory into the processor – the operand at LOCA is fetched and added to the contents of R0 – the resulting sum is stored in register R0.



Connection Between the Processor and the Memory

- 
- ▶ Add r4,r2,r3
 - ▶ $Pc \leftarrow MAR$
 - ▶ Issue read
 - ▶ Wait for the requested word to MDR
 - ▶ $Mdr \leftarrow IR$
 - ▶ $R4, R2 \leftarrow ALU$
 - ▶ ADDITION OPERATION
 - ▶ $ALU \leftarrow R3$
 - ▶ RESULT IN MDR
 - ▶ MAR
 - ▶ PC

Registers

- ▶ Instruction register (IR)
- ▶ Program counter (PC)
- ▶ General-purpose register ($R_0 - R_{n-1}$)
- ▶ Memory address register (MAR)
- ▶ Memory data register (MDR)

Typical Operating Steps

- ▶ Programs reside in the memory through input devices
- ▶ PC is set to point to the first instruction
- ▶ The contents of PC are transferred to MAR
- ▶ A Read signal is sent to the memory
- ▶ The first instruction is read out and loaded into MDR
- ▶ The contents of MDR are transferred to IR
- ▶ Decode and execute the instruction

Typical Operating Steps (Cont')

- ▶ Get operands for ALU
 - General-purpose register
 - Memory (address to MAR – Read – MDR to ALU)
- ▶ Perform operation in ALU
- ▶ Store the result back
 - To general-purpose register
 - To memory (address to MAR, result to MDR – Write)
- ▶ During the execution, PC is incremented to the next instruction

Interrupt

- ▶ Normal execution of programs may be preempted if some device requires urgent servicing.
- ▶ The normal execution of the current program must be interrupted – the device raises an *interrupt* signal.
- ▶ Interrupt-service routine
- ▶ Current system information backup and restore (PC, general-purpose registers, control information, specific information)

Bus Structures

- ▶ There are many ways to connect different parts inside a computer together.
- ▶ A group of lines that serves as a connecting path for several devices is called a *bus*.
- ▶ System bus-connects major computer components.


Address Bus-unidirectional

Data Bus-bidirectional

Control BIDIRECTIONAL Bus-regulates the activity sending control signals like memory read,memory write,I/O read I/O write,interrupt request,reset so on.

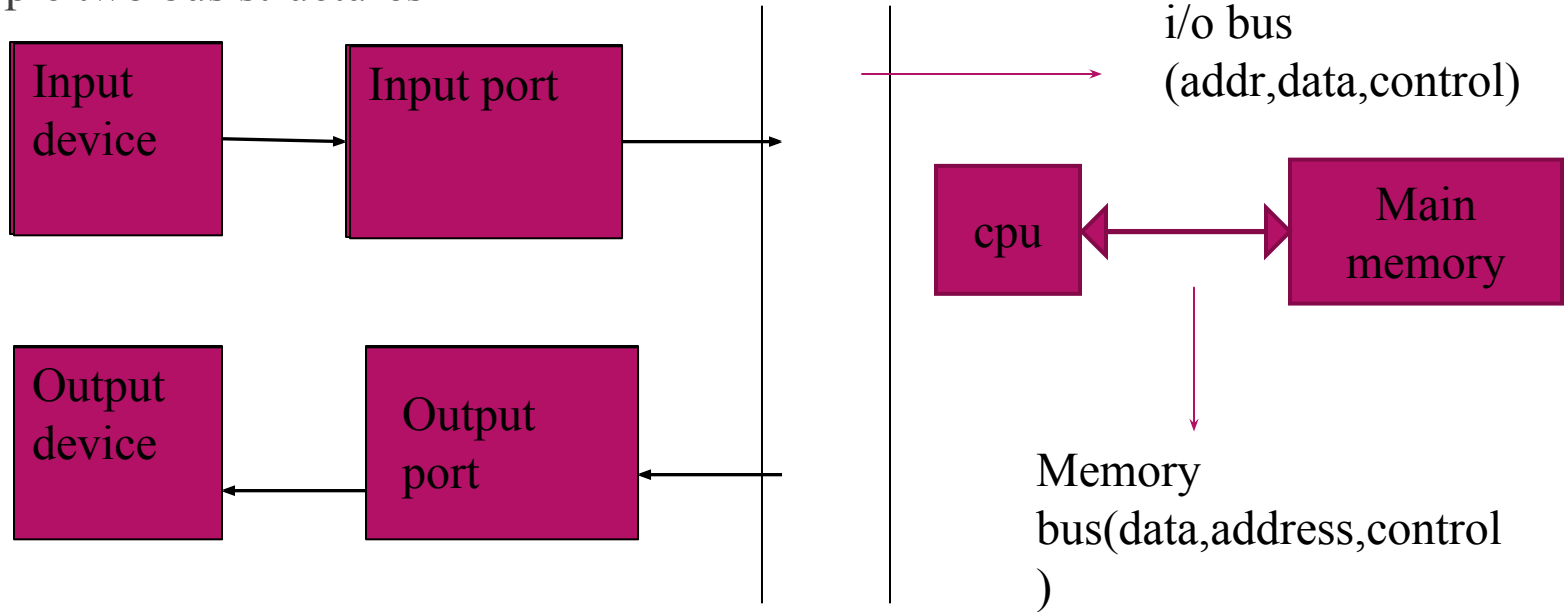
Bus Structure

- ▶ Single-bus

- 
- ▶ All units connected –system bus
 - ▶ Only 2 device communicate at a time
 - ▶ Advantage-Low cost,facility to connecting peripheral devices.
 - ▶ Disadvantage-slow speed.
 - ▶ Uses-minicomputers

Bus Structures

- Simple two bus structures



Speed Issue

- ▶ Different devices have different transfer/operate speed.
- ▶ If the speed of bus is bounded by the slowest device connected to it, the efficiency will be very low.
- ▶ How to solve this?
- ▶ A common approach – use buffers.

Performance

- ▶ The most important measure of a computer is how quickly it can execute programs.
- ▶ Three factors affect performance:
 - Hardware design
 - Instruction set
 - Compiler

Performance

- ▶ Processor time to execute a program depends on the hardware involved in the execution of individual machine instructions.

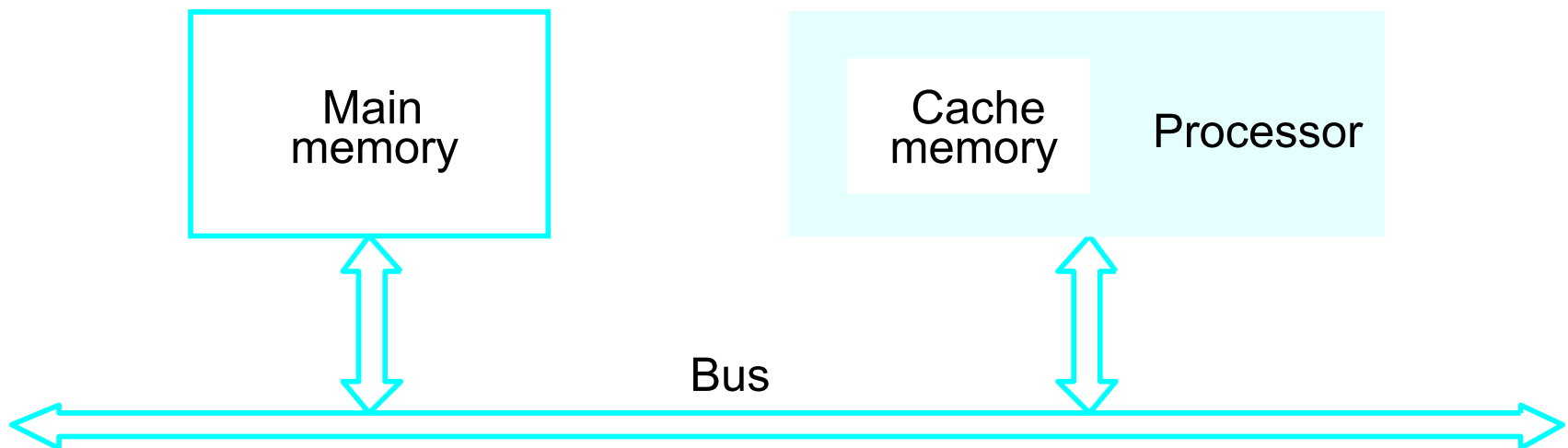


Figure The processor

Performance

- ▶ The processor and a relatively small cache memory can be fabricated on a single integrated circuit chip.
- ▶ Speed
- ▶ Cost
- ▶ Memory management

Processor Clock

- ▶ Clock, clock cycle, and clock rate
- ▶ The execution of each instruction is divided into several steps, each of which completes in one clock cycle.
- ▶ Hertz – cycles per second
- ▶ $\text{LENGTH OF CLOCK CYCLE} = P$
- ▶ $R(\text{CLOCK RATE}) = 1/P \text{ cps}$
- ▶ **HERTZ**

Basic Performance Equation

- ▶ **T** – processor time required to execute a program that has been prepared in high-level language
- ▶ **N** – number of actual machine language instructions needed to complete the execution (note: loop)
- ▶ **S** – average number of basic steps needed to execute one machine instruction. Each step completes in one clock cycle
- ▶ **R** – clock rate
- ▶ Note: these are not independent to each other

$$T = \frac{N \times S}{R}$$

How to improve T?

Clock Rate

- ▶ Increase clock rate
 - Improve the integrated-circuit (IC) technology to make the circuits faster
 - Reduce the amount of processing done in one basic step (however, this may increase the number of basic steps needed)
- ▶ Increases in R that are entirely caused by improvements in IC technology affect all aspects of the processor's operation equally except the time to access the main memory.

Instruction Rate

- ▶ Simple instruction requires small number
- ▶ Complex instruction requires large number
- ▶ Processor with simple instruction are called RISC
- ▶ Processor with complex instruction are called CISC
- ▶ Tradeoff between N and S
- ▶ A key consideration is the use of pipelining
 - S is close to 1 even though the number of basic steps per instruction may be considerably larger
 - It is much easier to implement efficient pipelining in processor with simple instruction sets
- ▶ Reduced Instruction Set Computers (RISC)
- ▶ Complex Instruction Set Computers (CISC)

Compiler

- ▶ A compiler translates a high-level language program into a sequence of machine instructions.
- ▶ To reduce N , we need a suitable machine instruction set and a compiler that makes good use of it.
- ▶ Goal – reduce $N \times S$
- ▶ A compiler may not be designed for a specific processor; however, a high-quality compiler is usually designed for, and with, a specific processor.

Performance Measurement

- ▶ T is difficult to compute.
- ▶ Measure computer performance using benchmark programs.
- ▶ System Performance Evaluation Corporation (SPEC) selects and publishes representative application programs for different application domains, together with test results for many commercially available computers.
- ▶ Compile and run (no simulation)
- ▶ Reference computer

$$SPEC\ rating = \frac{\text{Running time on the reference computer}}{\text{Running time on the computer under test}}$$

$$SPEC\ rating = \left(\prod_{i=1}^n SPEC_i \right)^{\frac{1}{n}}$$

PROBLEMS

- ▶ There are 5 Processors P1 is consuming $50\mu\text{sec}$ & every successive process consumes double the time of previous process. Calculate SPEC rating of each process & SPEC of entire suite(Assume reference system which can execute P1 in $100\mu\text{sec}$ & each successive process with increase of $50\mu\text{sec}$)

SOLUTION

- ▶ Number of processes=5
- ▶ $T_{exe}(P1)=50\mu\text{sec}$
- ▶ $T_{exe}(P2)=100\mu\text{sec}$
- ▶ $T_{exe}(P3)=200\mu\text{sec}$
- ▶ $T_{exe}(P4)=400\mu\text{sec}$
- ▶ $T_{exe}(P5)=800\mu\text{sec}$
- ▶ SPEC rating=Running time on the reference computer/Running time on computer under test

Continued...

- ▶ $\text{SPEC}(P1) = 100\mu\text{sec} / 50\mu\text{sec} = 2$
- ▶ $\text{SPEC}(P2) = 150\mu\text{sec} / 100\mu\text{sec} = 1.5$
- ▶ $\text{SPEC}(P3) = 200\mu\text{sec} / 200\mu\text{sec} = 1$
- ▶ $\text{SPEC}(P4) = 250\mu\text{sec} / 400\mu\text{sec} = 0.625$
- ▶ $\text{SPEC}(P5) = 300\mu\text{sec} / 800\mu\text{sec} = 0.375$
- ▶ $\text{SPEC}(\text{suite}) = 5\sqrt{2} * 1.5 * 0.625 * 0.375 = (0.703125)^{1/5}$
- ▶ $\text{SPEC}(\text{suite}) = \mathbf{0.932}$

Problem 2

- ▶ A program contains 1000 instructions. Out of that 25% instructions require 4 clock cycles, 40% instructions require 5 clock cycles & remaining require 3 clock cycles for execution. Find the total time required to execute the program running in a 1 GHz machine
- ▶ $T = (N * S / R)$

- ▶ $N=1000$
- ▶ 25% of $N=250$ instruction require 4 clock cycle
- ▶ 40% of $N=400$ instruction require 5 clock cycle
- ▶ 35% of $N=350$ instruction require 3 clock cycle
- ▶ $T=(N*S/R)$
- ▶ $4.05\mu\text{sec}$

Problem

- ▶ If the length of the clock cycle(P)= 0.8ns then calculate the clock rate
- ▶ $R = 1/P = 1/2 \times 10^{-9} = 500\text{MHz}$
- ▶ R is 500 million cycles per second

Multiprocessors and Multicomputers

► Multiprocessor computer

- Execute a number of different application tasks in parallel
- Execute subtasks of a single large task in parallel
- All processors have access to all of the memory – shared-memory multiprocessor
- Cost – processors, memory units, complex interconnection networks

► Multicomputers

- Each computer only have access to its own memory
- Exchange message via a communication network – message-passing multicomputers

Memory Locations, Addresses, and Operations

- ▶ Memory consists of many millions of storage cells, each of which can store 1 bit.
- ▶ Data is usually accessed in n -bit groups. n is called word length.

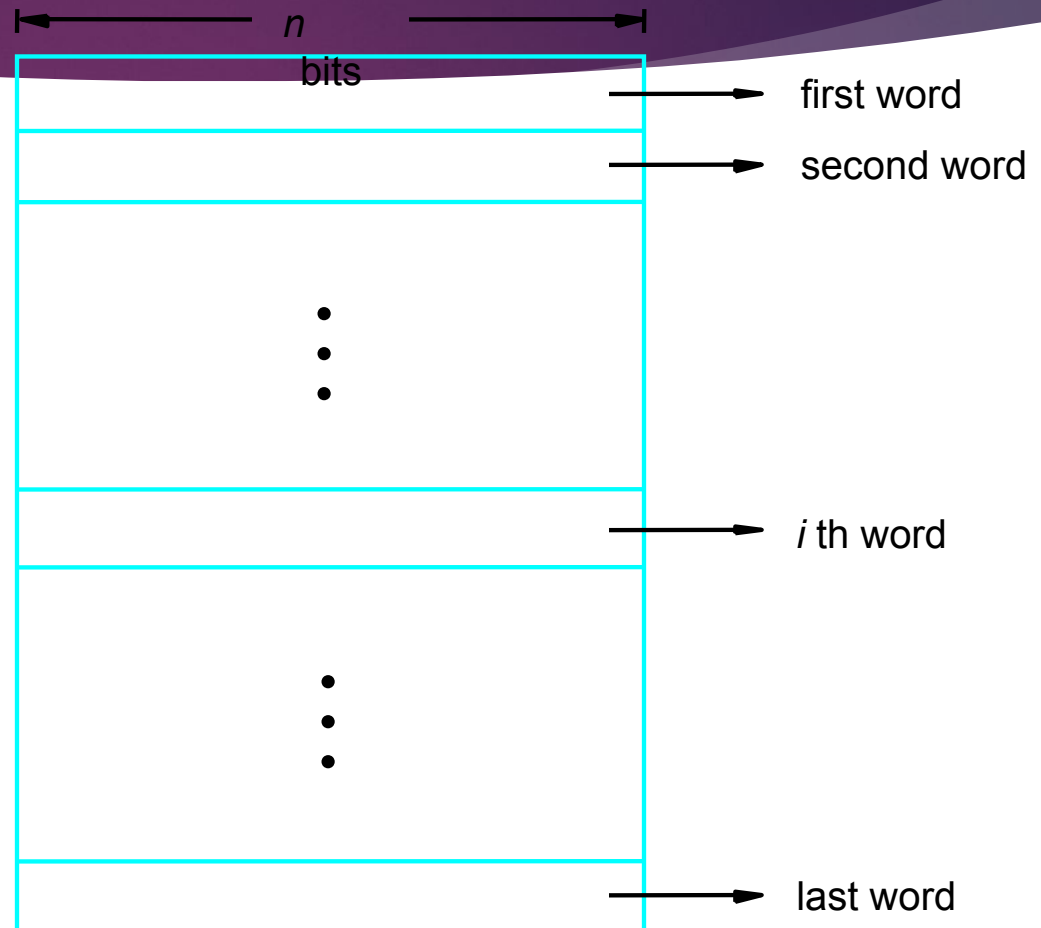
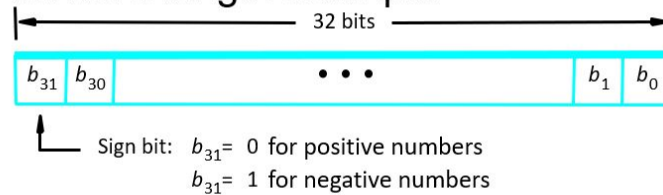


Figure 2.5. Memory words.

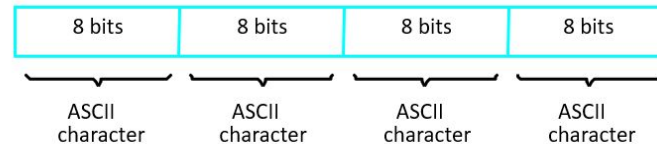
Memory Location, Addresses, and Operation

► 32-bit word length example

32-bit word length example



(a) A signed integer



(b) Four characters

Memory Location, Addresses, and Operation

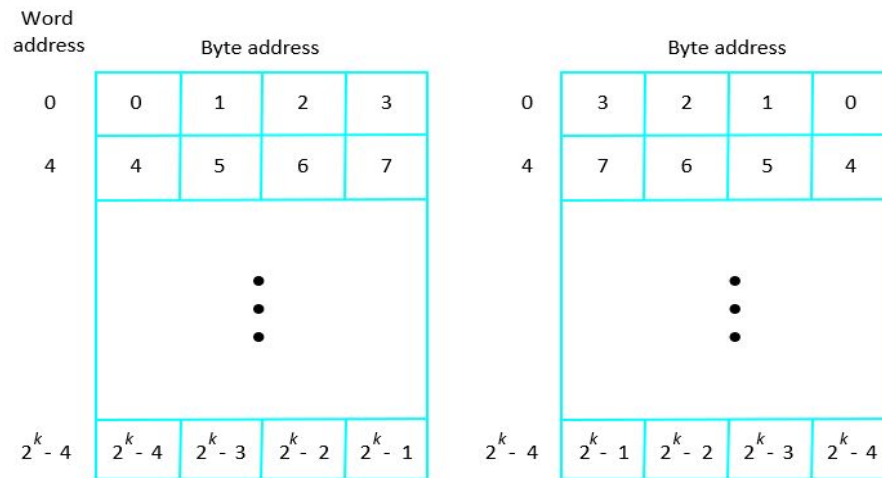
- ▶ To retrieve information from memory, either for one word or one byte (8-bit), addresses for each location are needed.
- ▶ A k -bit address memory has 2^k memory locations, namely 0 – $2^k - 1$, = 0 TO 15 called memory space.
- ▶ 24-bit memory: $2^{24} = 16,777,216 = 16\text{M}$ ($1\text{M} = 2^{20}$)
- ▶ 32-bit memory: $2^{32} = 4\text{G}$ ($1\text{G} = 2^{30}$)
- ▶ $1\text{K}(\text{kilo}) = 2^{10}$
- ▶ $1\text{T}(\text{tera}) = 2^{40}$

Memory Location, Addresses, and Operation

- ▶ It is impractical to assign distinct addresses to individual bit locations in the memory.
- ▶ The most practical assignment is to have successive addresses refer to successive byte locations in the memory – byte-addressable memory.
- ▶ Byte locations have addresses 0, 1, 2, ... If word length is 32 bits, they successive words are located at addresses 0, 4, 8,...

Big-Endian and Little-Endian Assignments

- ▶ Big-Endian: lower byte addresses are used for the most significant bytes of the word
- ▶ Little-Endian: opposite ordering. lower byte addresses are used for the less significant bytes of the word



(a) Big-endian assignment

(b) Little-endian assignment

Figure 2.7. Byte and word addressing.

Memory Location, Addresses, and Operation

- ▶ Address ordering of bytes
- ▶ Word alignment
 - ▶ Words are said to be aligned in memory if they begin at a byte addr. that is a multiple of the num of bytes in a word.
 - ▶ 16-bit word: word addresses: 0, 2, 4,....
 - ▶ 32-bit word: word addresses: 0, 4, 8,....
 - ▶ 64-bit word: word addresses: 0, 8,16,....
- ▶ Access numbers, characters, and character strings