Chapter 05 : Computer Architecture

5.0 Overview

Q : 05-00-01 : Describe Van Neumann's Stored Program Computer ?

Answer :

Van Neumann's Stored Program Computer : [A sequence of instructions (Program) and the data are stored in the memory of the machine. The machine reads the instructions one by one and executes these instructions accordingly.] In 1951, Van Neumann designed this stored program computer. This simple design proved to be powerful and general purpose. It is the basis of modern day computers.

Q : 05-00-02 : Describe most important components of modern stored program machine ?

Answer :

Control Unit (CU) : The control unit reads the instructions from the memory and decodes these instructions. This unit uses other components of the computer to execute the instructions given to the computer.

Arithmetic and Logical Unit (ALU) : General-purpose computer can perform different arithmetic operations on the data so it has a special unit that has electronic circuits to perform the basic arithmetic and logical operations on the data.

Main Memory : It is used to store program and data during execution. We call it working area of the computer or Primary Memory also.

I/O Unit : This handles the processor's communication with the peripherals (Disc drive, monitor, printer etc). There are registers to hold the data incoming or outgoing. Data and instructions should be in the memory before the computer can start executing, stored program computer uses I/O devices.

Bus Interconnection : The communication lines / channel used to connect different parts of the computer together.

The CPU : Central Processing Unit is the brain of the computer and is centrally located on the motherboard. The data come from the RAM (Random Access Memory) and the units (keyboard, drives etc.). After processing, the data is sent back to RAM and the units. The work consists mostly of calculations and data transport. Two typical components of a CPU are :

The Arithmetic Logic Unit (ALU) : The ALU is the part of computer that performs all arithmetic computations, such as addition and multiplication, and all comparison operations. The ALU is one component of the CPU.

The Control Unit (CU) : Which extracts instruction from memory, decodes and executes them, calling on the ALU when necessary.

Main Memory : A computer executes a program in its main memory. A computer cannot work without main memory in it. Mostly the modern computer memory is built in the form of a chip of a semi conductor material. It is built in the form of thousands or even millions of cells each capable of storing a bit i.e. 0 or 1. These cells are logically organized into group of 8 bits called a byte. Each byte in the memory has a unique number assigned, which is called the address of that byte. A memory is a sequence of bytes. CPU or any other component of the computer can access any byte from the main memory by specifying its address. Different bytes of the main memory can be accessed directly at random and very fast. There are two types of main memory :

RAM (Random Access Memory) : It is usually build by using two different technologies.

DRAM (Dynamic RAM) : Dynamic Random Access Memory, a type of memory used in most computers and needs an electric current to maintain electrical state.

SRAM (Static RAM) : In SRAM technology, the memory cells are made form digital gates and each cell can hold its value without any need to refresh the data as long as the power is supplied to it. As no refreshing is required by SRAM, these chips are faster than the DRAM chips and utilize less power. Its design is more complex and is more expensive. This memory is known as the cache memory and has a very small size. It increass the performance of a computer system and is volatile (unstable , the contents of the memory are lost as soon as the electricity supply is cut-off).

ROM (Read Only Memory) : It can be read but new data can not be written onto it so it is read only. The manufacturer of the ROM writes the data and programs permanently onto it and this data and programs cannot be changed afterwards. ROM contains frequently used instructions and data.

PROM (Programmable Read Only Memory) : Initially blank and the user or manufacturer can write data onto it by using special devices. Once the program / data is written onto PROM it can not be changed or altered.

EPROM (Erasable Programmable Read Only Memory) : Initially blank, programs and data can be written on it by the manufacturer or by the used by using special devices. Unlike PROM, the data written on it can be erased by using special devices (i.e. ultraviolet rays). So data / program written on it can be changed and new data can also be added on this form of ROM.

EEPROM (Electrically Erasable Programmable Read Only Memory) : This kind of ROM can be re-written by using electrical devices and so data stored on this ROM can be easily modified.

Important Note : All the forms of ROM are non-volatile so the data stored on these chips is not lost when electricity is cut-off. RAM is volatile i.e. when electricity is cut-off, all the data is lost.

5.1 Bus Interconnection

Q : 05-01-01 : Describe Bus Interconnection and explain three types of bus ?

Answer :

Bus Interconnection : [These components are interconnected by using a set of parallel (Conducting Wires).] Each of these lines can be used to transfer a sequence of bits from one component of the computer to the other component. This set of parallel lines is called BUS. The system bus is divided in three main categories :

Control Bus : [These lines are used to transmit different commands from one component to the other.]

Data Bus : [On the system bus 32 or 64 lines are reserved to transfer data from one component to the other.] A 64-line data bus can transfer 64-bits of data simultaneously so it is not difficult to see that the width of the data bus has a direct impact on the performance of the computer.

Address Bus : It is important to assign a unique ID (address) to each component. Not only the address is used to identify different components of a system but it is also used to specify different memory locations within the main memory.

Expansion Bus : As the number of components connected to the system bus increases, more components will be trying to use the system bus simultaneously. This will slow-down the computer as components will have to wait longer to get access to the bus. To solve this problem only the major components of the computer are connected to the system bus and remaining components are connected to another bus called expansion bus. The expansion bus is connected to the system bus.

5.2 The I/O Unit

Q : 05-02-01 : Describe I/O Unit ?

Answer :

The I/O Unit : We have many input / output devices like keyboard, mouse, disks etc. All these devices are very different from one another, can handle different data-transfer rates and support different data formats. It is impractical to connect all these devices directly to the system bus as these will take a lot of CPU time. A special hardware component I/O unit is used. Only the I/O unit is connected to the bus and the devices are connected to it. The I/O unit is responsible for keeping the track of states of different devices attached with it. It is also responsible for compensating the speed difference between the processor and the I/O devices.

Q : 05-02-02 : Explain the TWO ways / methods of transferring data from the peripherals into the computer ?

Answer :

Interrupts : The processor issues command to the I/O devices. When the devices get ready, these generate an interrupt signal for the processor. On sensing this signal, the processor suspends all other processing and performs the I/O operation. The disadvantage of this scheme is that it reduces the over all performance of the processor.

DMA (Direct Memory Access) : The processor issues I/O command and gets busy in other useful tasks. The special hardware gets the data from the I/O device and uses the system bus to place it in the main memory. When data is transferred, the processor does not need the system bus. So, the processor does not have to wait for the I/O operation to complete. The disadvantage of this scheme is that it is more complex and extensive and more hardware is needed.

Q : 05-02-03 : Explain CPU Registers ?

Answer :

CPU Registers : The program is stored in the main memory of the computer on contiguous (continuous) memory locations. Data is loaded into computer's memory before the processing starts and the control is given to the CPU. The CPU needs storage areas where data can be stored temporarily. As used frequently, so temporary storage areas are provided within the CPU for enhancing the performance efficiency of the CPU. These special purpose storage areas are called registers.

The Most Commonly Used CPU Registers

PC (Program Counter) : This register holds the address of the next instruction to be fetched for execution. As soon this instruction is fetched, its value is incremented so that it still has the address of next instruction.

IR (Instruction Register) : Once the instruction is fetched it is stored in the IR where this instruction is decoded.

MAR (Memory Address Register) : When the CPU wants to store some data in the memory or reads the data from the memory, it places address of the required memory location in MAR.

MBR (Memory Buffer Register) : The CPU uses this register to store data coming from the memory or going to the memory.

SP (Stack Pointer) : To understand the purpose of this register it is important to understand a very important data structure (Arrangement of Data) called Stack. This structure is an arranged storage in LIFO (Last in First Out) order.

GPR (General Purpose Registers) : These resisters are called EAX, EBX, ECX, EDX and can be used for any mathematical or logical operations. They are used for arithmetic and data movement purposes. Each can be divided in an upper and lower byte called AH, AL, BH, BL, CH, CL, DH, DI respectively. A stands for Accumulator, B for Base, C for Count and D for Data. Each of these registers can alternately be used as one byte, two byte or four byte registers, AH (1 byte), AX (2 bytes), EAX (4 bytes).

AX (Accumulator Register) : Used for arithmetic and data operations.

BX (Base Register) : Used for arithmetic and data movement and special addressing abilities.

CX (Counter Register) : Used for counting purpose. Acts as a counter for repeating or looping.

DX (Data) : Has special role in division and multiplication.

Address or Segment Registers : It is a group of 4, some times registers named CS, DS, ES and SS. The segment register is used as base location for program instructions, data, and the stack.

CS (Code Segment) : The CS register holds the base location of all executable instructions (code) in the program.

DS (Data Segment) : The DS register is the defaults base location for memory variables. The CPU calculates the offsets of variables using the current value of DS.

ES (Extra Segment) : The ES register is an additional base location for the memory variables.

SS (Stack Segment) : The SS register contains the base location of the current program stack.

Important Note : Each of the above registers has 2-bytes. These registers called segment register and are used in conjunction with either the IP register or two index registers DI and SI to address various areas of computer memory CS is the primary register or two index registers used to fetch instruction in conjunction with the IP register. DS is the primary register, used to point out data in the computer memory along with the DI or SI registers.

5.3 Computer Operations

Q : 05-03-01 : Explain different types of operations performed by computer ?

Answer :

Computer Operations : Different types of operations performed by the computer are :

Data Transfer Instructions : All CPUs provide different instructions for the transfer of data from and to the memory. A programmer can use these instructions to bring data into the CPU and copy data from the CPU to the main memory. Instructions have the following format.

Arithmetic and Logical Instructions : Another important category of operations, a CPU can do is Arithmetic and Logical Operations. Most CPU provides the basic arithmetic operations of add, subtract, multiply and divide for signed numbers and floating point numbers. Logical operations of comparing two numbers, performing XOR of number, shifting and rotating a number are some common form of logical operations provided by the CPU.

I/O Instructions : Every CPU provides its users with the operations of reading data from a peripheral device and writing data to a peripheral device. To use these operations a programmer may use input and print commands provided by the CPU.

Control Transfer : In all real world programs, given to the CPU must be repeated a number of times. To support such operations, all CPUs provide its programmers with control flow operations some examples of these operations are : Jump, Jumpz (Jump if zero) etc.

Instruction set : Each CPU provides its users with a number of instructions so that the users can perform different operation supported by the CPU. The set of all instructions provided by a CPU is commonly known as the instruction set of that CPU.

5.4 The Instruction Format

Q : 05-04-01 : Define Instruction Format ? Explain various Instruction Code Formats in use ?

Answer :

The Instruction Format : A computer will usually have a variety of instruction code formats. It is the function of the control unit within the, CPU to interpret each instruction code and provide the necessary control functions needed to process the instruction. Each instruction for the CPU is specified by giving : “A Code For the Instruction (opcode)” and “Addresses of the Operands”.

Zero-Address Instruction Format : The name “Zero-Address” is given to this type of computer because of the absence of an address field in the computational instructions. A stack-organized computer does not use an address field for the instructions ADD and MUL. The PUSH and POP instruction, however need an address field to specify the operand that communicates with the stack.

One-Address Instruction Format : One-address instructions use an implied accumulator (AC) register for all data manipulation. For multiplication and division there is a need for a second register.

Two-Address Instruction Format : For two address instructions, each address field can again specify either a possible register or a memory address. Two-address instructions are the most common in commercial computers. Examples of such instructions are MOV, ADD, CMP and BIS.

Three Address Instruction : Computer with three-address instruction formats can use each address field to specify either a processor register or memory operand. The advantage of the three-address format is that it results in short programs when evaluating arithmetic expression. The disadvantage is that the binary-coded instructions require too many bits to specify three addresses.) The instruction formats in the computer are restricted to either three register address fields or two register address fields and one memory address field.

Q : 05-04-02 : Explain Fetch-Decode-Execute Cycle with diagram ?

Answer :

Fetch-Decode-Execute Cycle : When we want to execute a sequence of instructions (program) those instructions / data are first of all loaded into the main memory of the computer by using some I/O device. Once these instructions have been loaded into main memory, the address of the first instruction is copied into the program counter and the control is given to the CPU. The CPU performs the following steps :

Fetch Instruction : The CPU reads the value of PC and the instruction pointed to by PC into the instruction register. This instruction involves the following steps :

Copy the contents of PC into the MAR and request a memory read.

Copy the data read from the memory into MBR and instruction in the IR. Increment PC, so that it points to the next instruction.

Decode Instruction : Once the fetching of instruction is complete the CU decodes the instruction by analyzing the opcode of the instruction. It also reads the values of operands specified in the instruction. Decoding means activating the appropriate circuit to execute the instruction.

Execute Instruction : After decoding the instruction (the processor executes the instruction by using the activated circuit. Then the results of the execution are written back to registers and memory.

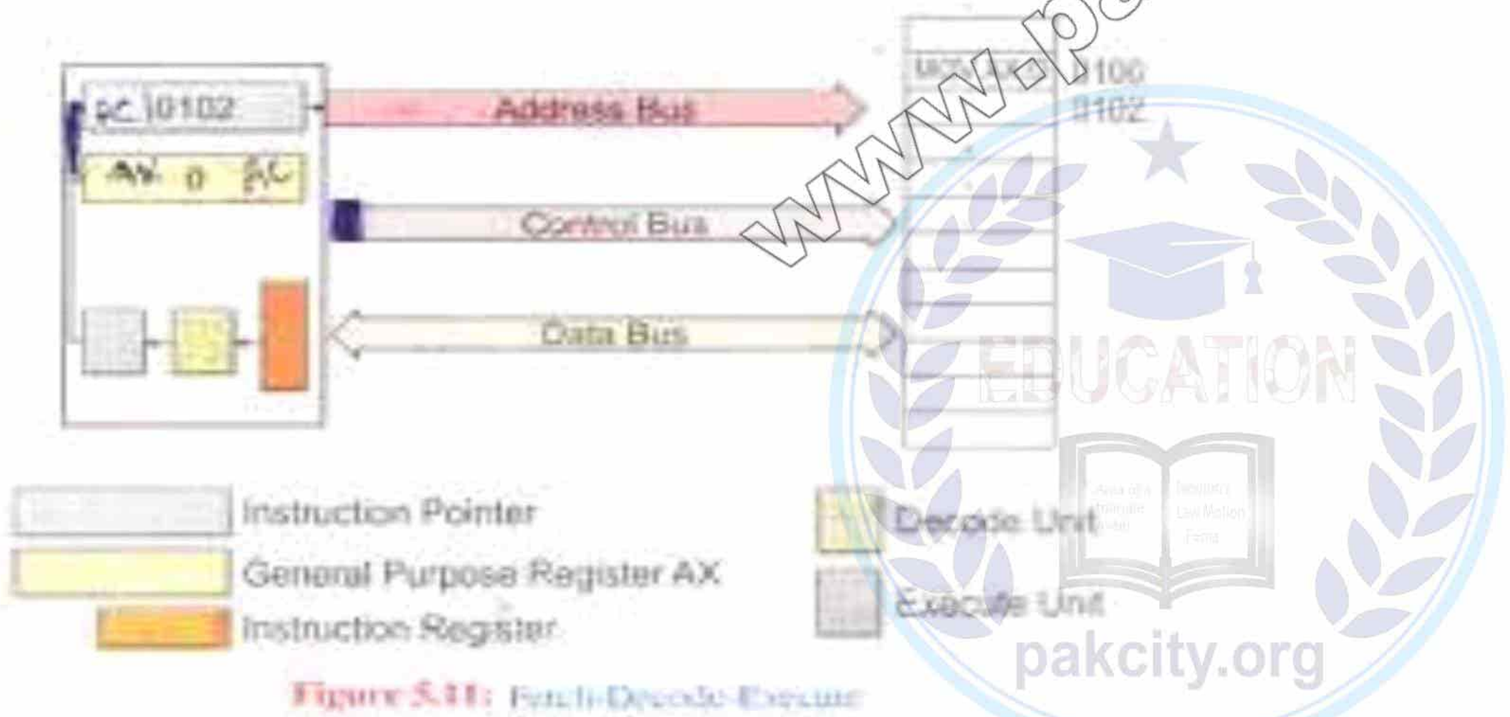

Figure 5.11: Fetch-Decode-Execute

5.5 Software

Q : 05-05-01 : Define Software ? Differentiate between Application Software and System Software ?

Answer :

Software : The term software is used for [a sequence of instructions given to the computer to perform a specific task]. Software consists of the Step-By-Step instructions that tell the computer what to do.

Application Software : Software that can perform useful work on general purpose tasks. It may be either customized (special order) or packaged (ready made).

Customized Software : It is a software designed for a particular customer. The program can be developed by a single computer professional programmer or by a team of programmers depending upon the requirements.

Package Software : It is the kind of, Off-The-Shelf (ready made) program developed for sale to the general public. Package software includes word processing, spreadsheet, database manager, graphics, and communications programs.

System Software : System software, which includes operating system and device drivers, enables application software to interact with the computer.

Operating Systems : It is obvious that to solve some problem on a computer a programmer will write instructions. But other than writing instructions for solving the problem every programmer will also have to write instructions for the following tasks :
Read data from the input devices and show results on the output devices.
Perform memory management tasks (more details latter).
Organize data on the storage devices.
These tasks are very complex and only expert programmers can write these instructions. OS [a set of programs running in the background on a computer system and providing an environment in which other programs can be executed and the computer system can be used efficiently.]

Q : 05-05-02 : Explain Main Functions of Operating System ?

Answer :

Main Functions of Operating System

Manage Hardware Resources : The operating system must provide programmer for managing the hardware resources of the computer like disks, memory, and CPU.

Memory Management : Every program is to be loaded into the computer's main memory during the execution. If there are several programs simultaneously loaded in the main memory, as is the case when time-sharing is used, the program and its data must be protected from the actions of other programs.

Load and Execute Programs : A program has to be loaded into the main memory before the processor can execute it. The OS provides the facility of easily loading a program into memory and starts its execution.

Data Security : The OS must also protect the user data against illegal access and modification.

Providing Interface to The Users : The OS must provide an interface between the user and the computer and also between software and the computer. Types of interfaces :

Command Prompt : Users communicate with the operating system by typed commands using a keyboard. Each command given to the OS activates one of the many programs in the OS.

Graphical User Interface (GUI) : The GUI interface consists of Window, Menus, Icons and pointers. The user of the system communicates with the OS by selecting commands from the menus or by selecting different icons with the pointing devices.

5.6 The Translators and Their Functions - Interpreters and Compilers

Q : 05-06-01 : Describe Machine Code, Assembler, Assembly Language, Compiler, Source Code and Object Code and Interpreter ?

Answer :

Machine Code : [The machine code instructions are the instructions provided as the instruction set of the machine and these codes are represented by a binary pattern in the computer]. To write a program the programmer had to write the instructions in binary. This was a very complex task and even writing very simple programs took a long time. It was very difficult to produce correct programs as detecting errors in the programs and correcting these errors (Debugging the program) was very difficult.

Assembler : [A program that produces the binary instruction for a given assembly language program is called an Assembler].

Assembly Language : Assembly is a low level language, which provides instruction set for a particular machine. Each assembly instruction maps to a single machine instruction so it is very easy to translate a program written in assembly language to machine code. Writing programs in assembly language is very easy but is still a tedious work and took a long time.

Compiler : [The program that translates a high-level language program into machine language is called a Compiler]. A more appropriate definition of a compiler is that [it is a program that takes as input a high-level language program and generates an object program]. This object program may or may not be the absolute (final) machine code.

Source Code and Object Code : Once a program has been translated into machine code it can then be loaded into the main memory and executed by the CPU. [The high-level language version of the program is usually called the Source Code and the resulting machine code program is called the Object Code].

Interpreter : Another useful translator is an interpreter. An interpreter takes as input a high-level language program and performs the following actions :

It repeatedly reads instructions (one at a time) and translates it to machine code.

It then executes the instruction.

Q : 05-06-02 : Differentiate between Compiler and Interpreter ?

Answer :

Difference Between Compiler and Interpreter : One difference between a compiler and an interpreter is that a compiler converts each instruction only once but an interpreter may translate an instruction several times. Clearly if an instruction has some error an interpreter can easily identify it. Also an interpreted program runs slower than a compiled program as once a program is compiled it does not need any further translation but the original program has to be translated every time it is executed by an interpreter. And some instructions will be translated several times.