Syllabus - computer organisation and architecture

I Basic computer organisation and design
Operational concepts, instruction code, computer registers, computer instructions, memory locations and addresses, instruction cycle, timing and control, business organisation

II CPU
General register organisation, addressing modes, stack organisation, instruction classification, program control ?

III Memory Organisation

Memory hierarchy, main memory, organisation of RAM, SRAM and DRAM, ROM, P-ROM, EPROM, EEPROM, auxiliary memory, cache memory, virtual memory, memory mapping techniques.

IV Parallel computer structures

Introduction to parallel ~~computer structures~~. processing, pipeline computer, multiprocessor system, architectural classification schemes - SISD, SIMD, MISD, MIMD.

V Pipelining and Vector Processing

Introduction to pipelining, instruction and arithmetic pipelines, vector processing, array processors.

03/01/2020

UNIT - I

Basic Structure of Computers

Computer organisation deals with functional designing of various units of a digital computer.

Computer architecture deals with various operational functions of hardware units and the method of information handling inside the computer.

Functional units of a Computer

A computer system consists of five functional units.
Input unit, Memory unit, ALU - Arithmetic Logic Unit, Output Unit and Control Unit.

Input unit accepts information from the user and it is either stored in computer memory or in the processor to perform desired operation. The information can be classified into two category:-

- Instructions
- Data

Instructions are commands that govern the transfer of information within a computer as well as between computer and input-output devices. The instruction can be a command

specifying arithmetic or logic operations to be performed. A set of instructions that perform a task is called program

Data are numbers and encountered character that are used as operands by the instructions.

Input Unit

Computer accepts data through input units In a keyboard whenever a key is pressed, corresponding letter or digit is automatically translated into its corresponding binary code.

eg:- keyboard, web camera, scanner

Memory Unit

The function of a memory unit is to store data. There are two types of storage:-

- primary unit
- Secondary unit

Primary storage is fast and expensive. Programs ~~Program~~ must be stored into this memory while they are being executed.

Secondary memory are slow and cheaper as compared to primary memory.
Large amount of data and many program

can be stored in secondary memory for future use.

Arithmetic Logic Unit (ALU)

All the arithmetic and logical operations are performed in this unit. The result obtained from the ALU are either stored in the memory or stored within the processor itself for immediate use.

The arithmetic operation is performed by the ALU when the operands are brought into the processor. There are high speed internal storage elements within the processor called registers.

Output Unit

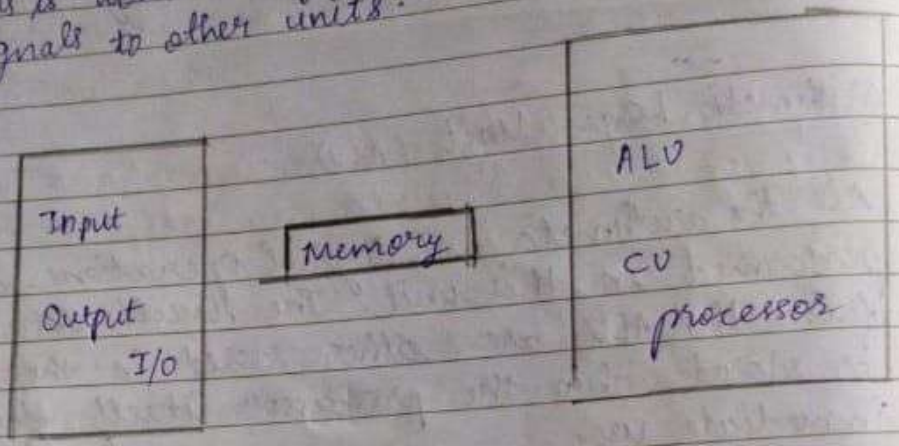
The output unit is the counter part of the input unit. It's function is to send processed results to the outside world.

eg:- monitor, printer, speaker

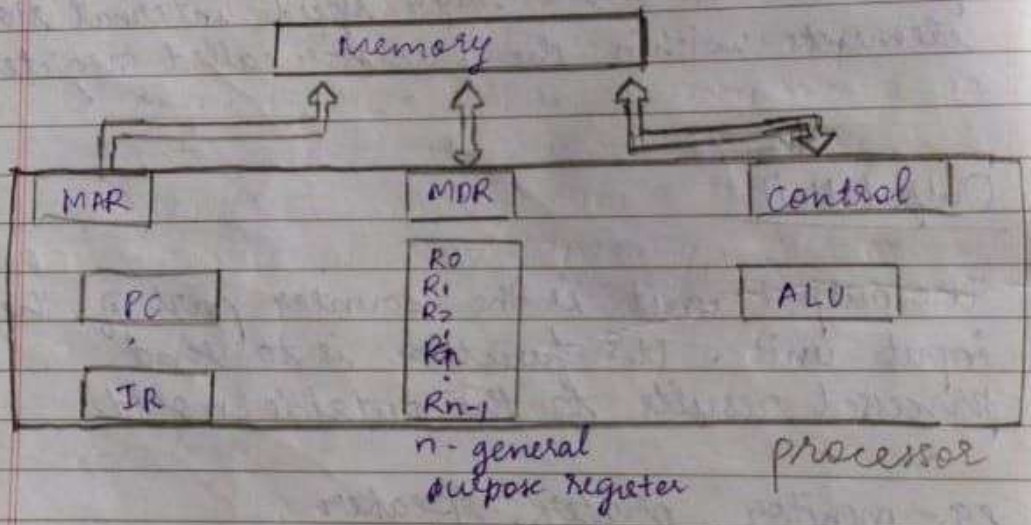
Control Unit

The control unit is known as the nerve system of a computer. It co-ordinates all other parts for performing the desired operations

This is achieved by assuring proper timing signals to other units.



Basic Operational Functions



The above figure shows various purpose registers inside the processor and the connections between the processor and the memory.

The instruction register (IR) is a special purpose register. that hold the instruction that currently being executed.

memory → processor (Read)
processor → memory (write)

The program counter (PC) is a specialized register that keeps the address of the next instruction to be executed. The content of the PC is updated after execution of every instruction There are two registers that help the communication with the memory (MAR and MDR)

MAR (Memory Address Register) holds the address of the location to be accessed.

MDR (memory Data Register) contains the data to be written into or read out of the address location.

Consider the instruction, ADD A, R0 the execution of the program starts when the PC is said to pointing the first instruction of the program. Considering the above example the content of PC is transferred to MAR, and the read control signal is sent to the memory.

After the time required to access the memory the particular addressed word is read out from the memory and loaded into MDR. The content of MDR is then transferred to IR. At this particular the instruction is ready to decode and execute.

All the above example the operand are residing inside the memory location and the register. The operands has to be fetched by sending its address to MAR and initiating a read

Execution cycle (instruction reading)
1. Fetching instruction from memory to primary memory
2. Decode (convert to machine form)
3. Execute

because cycle. When the operands are reading from memory to MDR it is transferred from MDR to ALU.

After fetching the two operands in this way ALU performs the desired operation and the result is send to MDR. The address of the location where the result is stored must be send to MAR and the write signal is initiated.

Bus Structure

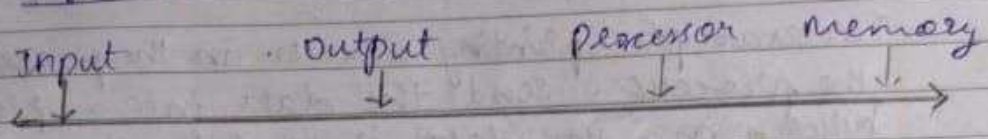
A group of lines that serves as a connecting path for several devices of computer is called a bus. A bus is organised in such as way that when a word of data is transferred between units all its bits are transferred simultaneously over many lines, one per line. The most commonly used bus system is a single bus system in which all the units are connected to a single bus.

The main disadvantage is that the bus can only use only one transfer at a time; that is only two units can actively use the bus at a given time.

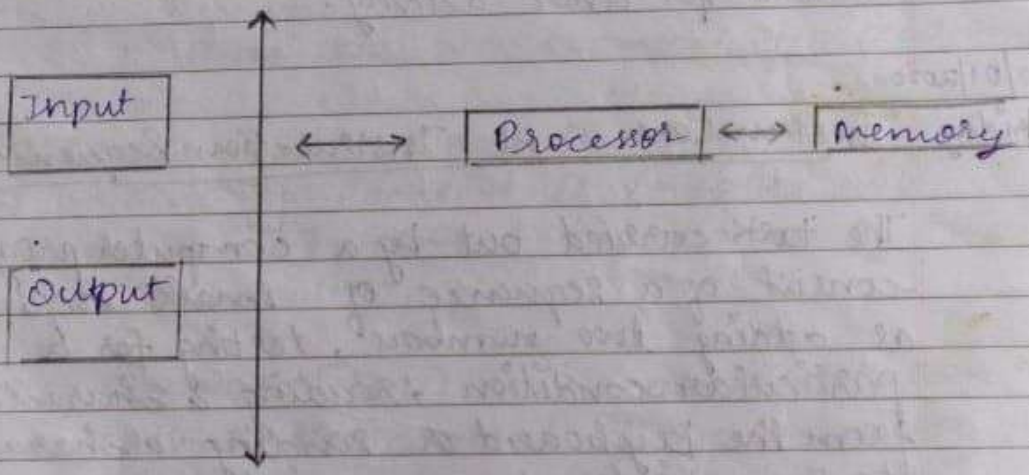
The main advantage of a single bus structure is low cost and flexibility for attaching

periferal devices.

Single bus structure



Two way Bus Structure



The two way bus structure containing input output bus (I/O) and memory bus. The input, output unit and processor is directly connected to input output bus.

The bus which connect and memory is called memory bus. The various device that connected to a bus is different in

the speed of operation. The transfer mechanism is used to smooth out the difference in timing among processors and memory. An external device is called buffer register.

For example printing a data in the processor the processor send the data into a print buffer, once the data is loaded the printer can start printing; without further intervention by the processor. The bus and the processor are no longer needed and can be released for other activity.

10/01/2020
Friday

Instructions and Instruction Sequencing

The task carried out by a computer program consist of a sequence of small steps such as adding two numbers, testing for a particular condition, reading a character from the keyboard or sending a character to be displayed on a display screen.

A computer must have instructions capable of performing four types of operations

(1) Data transfer between memory and processor registers

(2) Arithmetic and logic operations on data

(3) Program sequencing and control

(4) Input output transfer

Register transfer notations

The instructions can be stored in the memory or the processor register.
The name of each location can be addressed using a variable or a number.

LOCK, A, Place, etc are examples of location names.
Registers can be represented using $R_0, R_1, R_2 \ldots$
The content of a location is denoted by placing a [] around the name of the location.

for eg:- $R_1 \leftarrow [LOC]$. The above notation means that the content of the memory location [LOC] is transferred into the processor register R_1.

eg:- $R_3 \leftarrow [R_1] + [R_2]$. The content of R_1 and processor register R_1 and R_2 is added and the result is transferred to the processor register R_3.

This type of notations are known as register transfer notations or (RTN)

The right hand side of an RTN always denotes a value and the left hand side is the name of the location where the value is to be placed, overwriting the old contents of that location.

Assembly Language Notations

We use another type of notation to represent machine instructions and programs. For this purpose we use an assembly language format. For example: an instruction that causes the transfer from the memory location LOC to the processor register R_1 is specified by the statement

MOVE LOC, R_1

The content of LOC is unchanged, a copy of it will be loaded by the execution of this instruction, but the old content of Register R_1 is overwritten

eg:- ADD R_1, R_2, R_3 ie = $R_3 \leftarrow (R_1) + [R_2]$

The above notation is used for adding two numbers ~~R_1 and R_2~~ contained in processor registers R_1 and R_2 and placing their sum in R_3.

Instruction Types - (Basic Instruction Type)

The instruction types are of three:

(1) Three address instruction
(2) Two address instruction
(3) One address instruction

Three address instruction

The operation of adding two numbers using the statement C = A + B can be explained using all the three categories of instruction types.
In order to add the values of memory location A and B and storing the result in C, you can use the three address instruction as follows:-

ADD A, B, C

Operands A and B are called the source operands and C is called the destination operand.

Add is the operation to be performed on the operands.

The general instruction of this type can be written as :- operation source 1, source 2, destination.
eg:- ADD A, B, C
ie [C] ← [A] + [B]

Two address instruction

In this type of instructions each instruction will be having only two operands :- general form is

operation source, destination

eg:- ADD A B

ie [B] ← [A] + [B]

MOVE B, C ie [C] ← [B]

One Address Instruction

When a machine instruction specify only one memory operand along with the operation, it is known as one address instruction.

When a second operand is needed, it is understood implicitly to be in an unique location - a processor register called accumulator.

To perform the statement [C] = A + B using a one address instruction, we need to perform the following sequence of instructions

eg:- Load A [Acc] ← [A]

Add B [Acc] ← [Acc] + [B]

Store C [C] ← [Acc]

The general form is :-

Operation source/destination.

17/01/2020
Friday

Instruction Execution and Straight Line Sequencing

Consider the word length of memory is 32 bits and byte address abling. The three instruction of the program are in successive memory location. Since each instruction is 4 byte long, the second and the third instruction start at the address i+4 and address i+8.

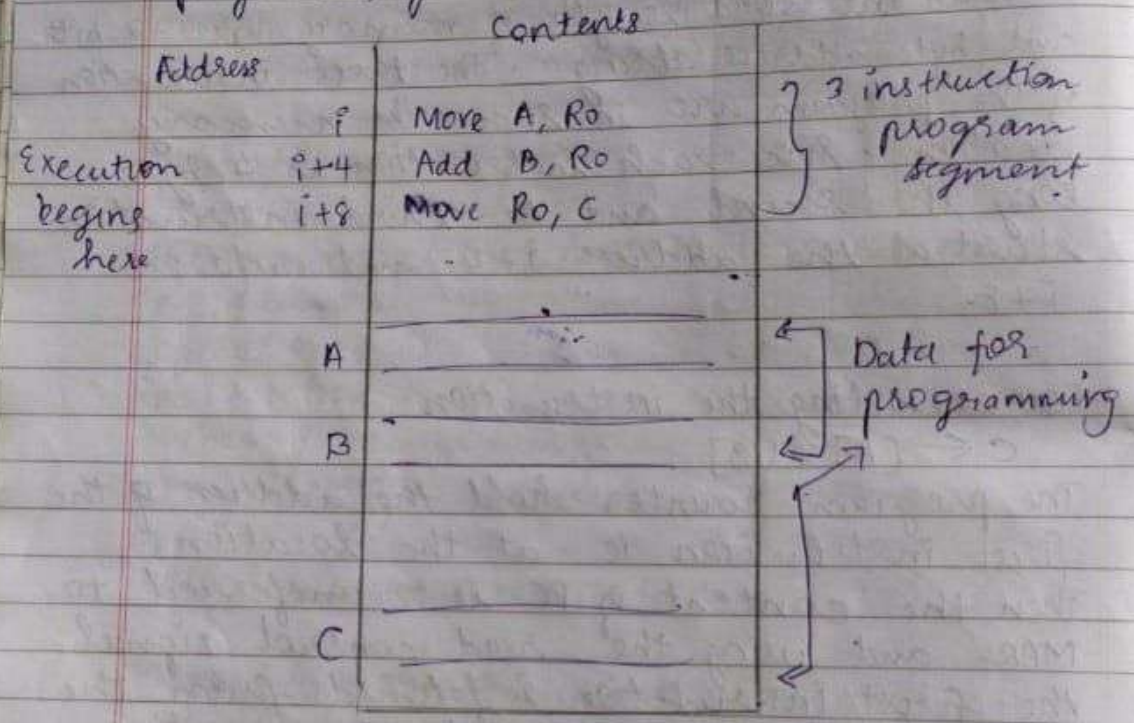
For executing the instruction

$C \leftarrow [A] + [B]$

The program counter hold the address of the first instruction ie at the location i. Then the content of PC is transferred to MAR and using the read control signal the first instruction is fetched from the memory and stored in the instruction register. After fetching the first instruction the content of PC is incremented to the next location.

The process of fetching and executing one at a time, in the order of increasing address is called Straight line sequencing.

During the execution of each instruction the PC is incremented by 4 to point next instruction. After the MOVE instruction at location i+8 is executed, the PC contains the value i+12 which is the address of the first instruction of the next program segment.



A program for C ← (A) + (B)

Executing an instruction is a second face procedure. The first face procedure is called instruction fetch.

Instruction is fetched from memory location and placed in instruction register.

The second face is called instruction

execution. The instruction in the IR is examined to determine the operation to be performed. This involves fetching operand from the memory or from the processor registers, performing arithmetic and logical operations and stored the result in the destination location.

At some point during the second face procedure the content of the PC are advanced to point to the next instruction that when they execute face of an instruction is completed; the PC contains the address of next instruction and a new instruction fetch face can begin.

Generating Memory Addresses

The memory operands addresses cannot be given directly in a single instruction in the loop. Otherwise it would need to be modified on each pass through the loop. Suppose that a processor register, Ri are used to hold the memory addresses of an operand. If it is initially loaded with the addresses num1 before the loop is entered; and is then incremented by 4 on each pass through the loop, it can be provided with the needed capability.

To specify an address of an operand we use

addressing modes. The different ways in which the location of an operand is specified in an instruction are referred to as addressing modes.

~~28/01/2020~~ ~~Tuesday~~

Computer organization

- concerned with the way the hardware components are connected together to form a computer system.
- It is defined by
  - → its internal registers
  - → the timing and control structure and
  - → the set of instructions that it uses