Aug 2; 24

UNIT- I

Ques What is Computer Architecture?

Ans- C.A. can be defined as a set of rules and methods that describe the functionality, management and implementation of computers. It is nothing but rules by which a system performs and operates.

Sub-divisions → ① Instruction Set Architecture or I.S.A

① I.S.A:-
Whenever an Instruction is given to processor its role is to read and act accordingly.

② Micro-Architecture:-
It describes how a particular processor will handle and implement Instructions from I.S.A.

③ System Design:-
It includes hardware components within the system.

XTRA EDGE .....
Date : / /
Page :

Role of Computer Architecture

The main role of C.A. is to balance the performance, efficiency, cost and reliability of a computer system.

for eg. → I.S.A. act as a bridge b/w computer's software and hardware.

Computer can only understands binary language (0,1) and user understands high level lang.; so to communicate b/w user and computer, ISA plays a major role.

① Processor → ALU, Control Unit, Bus interface
② Memory → Memory Array, Address Decoder
③ Peripheral → Peripheral Logic, Address Decoder

Aug 5, 24

Register Transfer Language

A digital computer system exhibit an interconnection of digital modules such as registers, decoders, arithematic elements and control logic.

These digital modules are interconnected with some common data and control path to form a complete digital system. Moreover, digital modules are best define by the registers and operations that are performed on the data stored in them.

(2) 1M

Operations performed on the data stored in registers are known as micro-operations.

The internal hardware organisation of a digital system is best defined by:

1. The set of registers and the flow of data b/w them.
2. The sequence of micro-operations performed on the data which are stored in the registers.
3. The control path that initiates the sequence of micro-operations.

→ The Register Transfer Language is the symbolic representations of notations used to specify the sequence of micro-operations.

In a computer system, data transfer takes place b/w processor, registers and memory as well as b/w processor, registers and Input - Output system. These data transfer can be represented by standard notations like

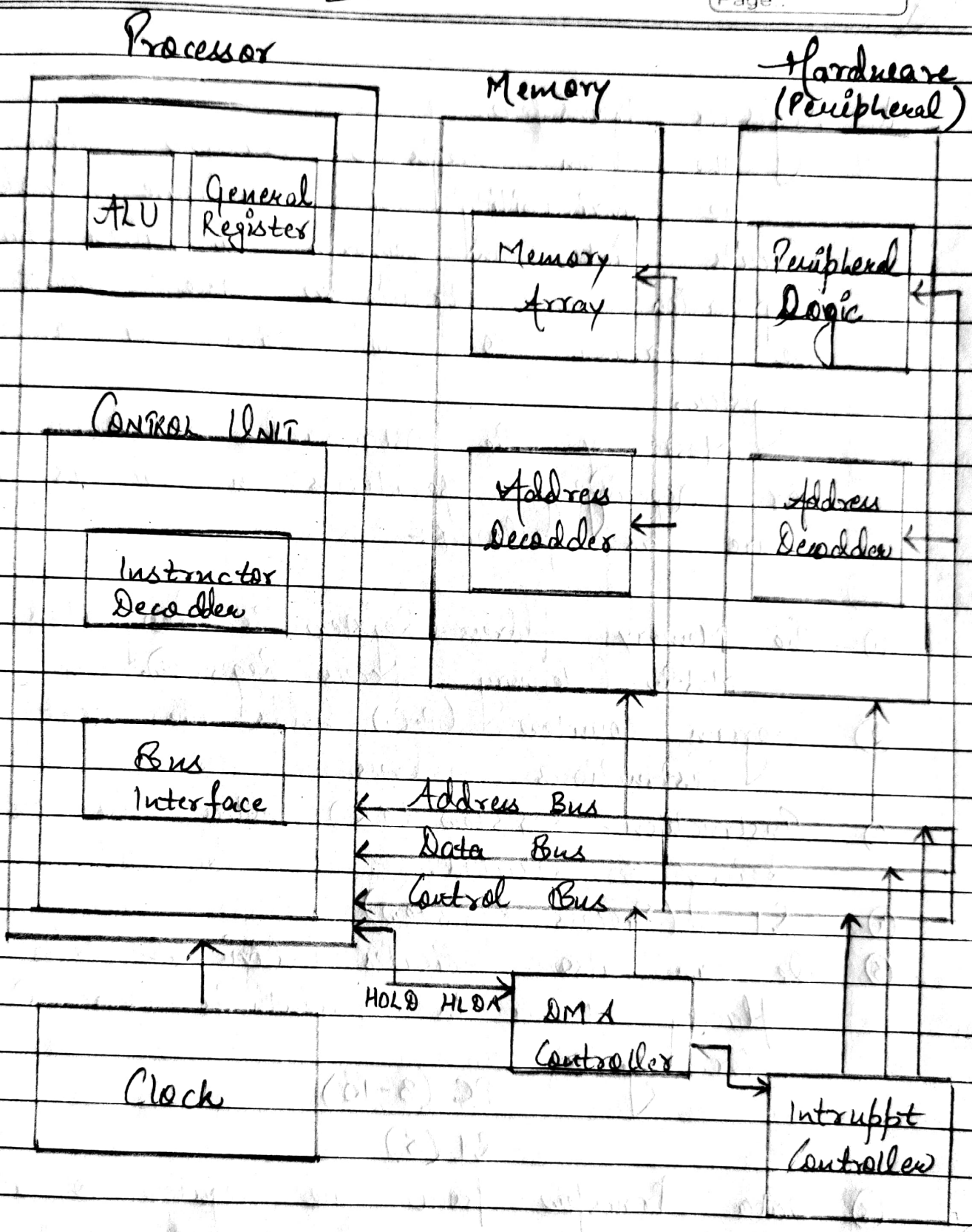
1. R0, R1, R2, ---. represent processor registers.

2. The address of memory locations (~~Place~~) are represented by names such as PLACE, MEM and LOC etc. (lateral Description complex)

3. I-O registers are represented by name such as ~~data~~ DATA-IN, DATA-OUT.

4. The content of register of memory is denoted by placing [ ] around the name of registers and memory locations.

C.t.c



DMA → Direct Memory Access

Aug 8:24

Date : / /
Page :

Register Transfer

The term register transfer refers to the availability of hardware logic circuits that can perform a given micro-operation and transfer the result of the operation to the same or another register.

Most of the standard notations use for specifying operations on various registers are as follows:-

1. The Memory Address Register is designated by M.A.R. (Memory Access Register).
2. Program Counter (P.C.) holds the next instructions address.
3. Instruction Register (I.R.) holds the instruction being executed.
4. R1 (Process Register)
5. We can also indicates individual bits by placing them in parenthesis (brackets) for eg →

   PC (8-10)

   R1 (5)
6. Data Transfer from one register to another register is represented in symbolic form by means of Replacement Operator.

   for eg →

   R2 ← R1
7. Typically most of the users want to transfer only in a pre-determined control conditions.

But we also use if --- then statement.
for eg. →

If P=1 then R2 ← R1
↳ (Program Counter, Processor)

Aug 13:24

Bus and Memory Transfer

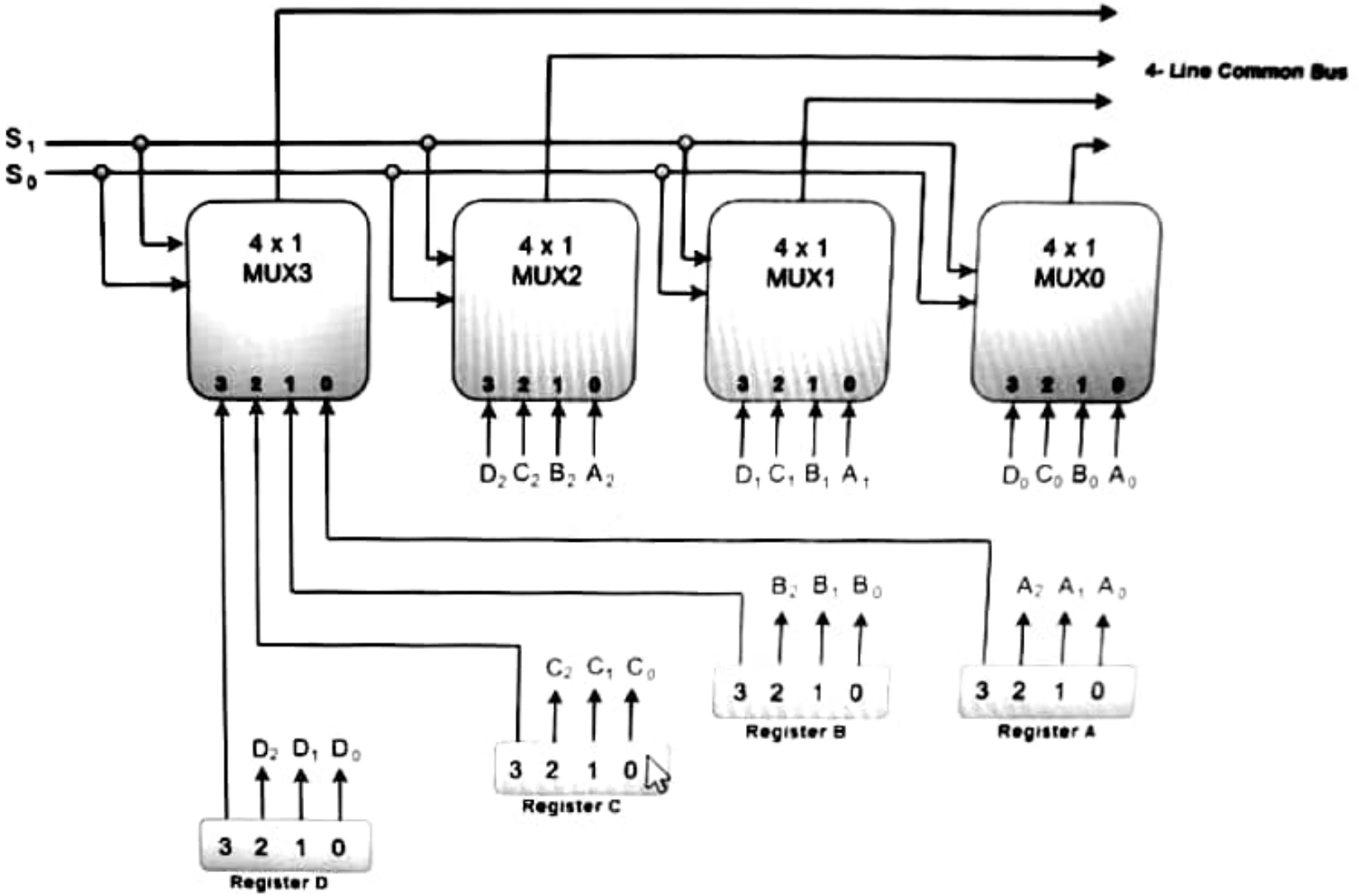
A digital system composed of many registers and path must be provided to transfer information from one register to another.
The number of wires are connecting all of the registers will be excessive if seperate lines are used b/w each registers and all other registers in the system.

Bus - A bus structure, on the other hand is more efficient for transfering information b/w registers in a multi-register configuration system.
A bus consists of a set of common lines one for each bit of register through which binary information is transfer one at a time.

Control signal determines which register is selected by the bus during a particular register transfer.

of a Bus system for four registers. For instance, output 1 of register A is connected to inpu

Bus System for 4 Registers:



The two selection lines S1 and S2 are connected to the selection inputs of all four multiple selection lines choose the four bits of one register and transfer them into the four-line common

electronic circuits which are used to transfer the signals through bus.

S1	S2	R
0	0	A
0	1	B
1	0	C
1	1	D

The two selection lines S1 and S2 are connected to the selection I/P of all four multiplexers. The selection lines chose the 4 bits of one register and transfer them into the 4 line common bus. When both of the select lines are at low logic, i.e., S1, S2 = 0, 0; the zero data I/P of all 4 multiplexers are selected and applied to the outputs that form the bus.

Similarly, when S1, S2 = 0, 1; register B is selected and the bus will receive the content provided by register B and so on.

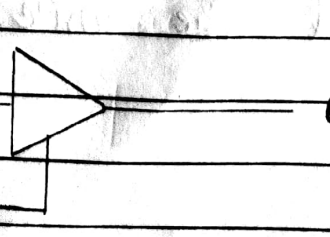
Aug 14; 24

3-State Bus Buffer

The three state gates can be considered as a digital circuit that has 3 gates. Two of which are signals equivalent to logic 1 and 0 as in a conventional gate. However, the third gate exhibit a high impedance state.

The most commonly used 3-state gates in case of bus system. It is known as buffer gate.

The graphical symbol is represented



- $Y = A$, if $C = 1$
- High impedance, if $C = 0$

Memory Transfer

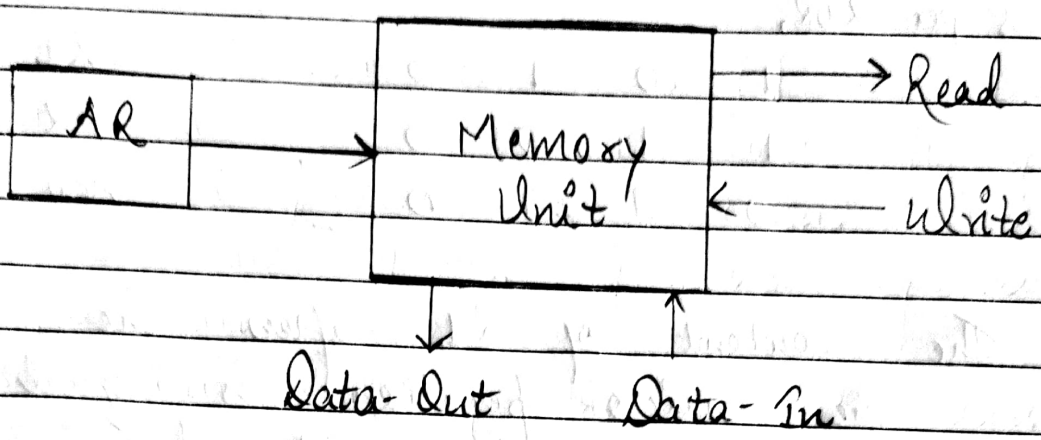
Memory Transfer notations are used as follows:

1. The transfer of information from a memory to the user is called a read operation.
2. The transfer of new information to be stored in the memory is called write operation.
3. A memory word is designated by the letter 'M'.
4. We must specify the address of memory word while writing the memory transfer operation.
5. The address register is designated 'AR' and the data register by 'DR'.
6. Thus, a read operation can be stated as

$$\text{Read}: M[AR \leftarrow R1]$$

and the write operation as

$$\text{Write}: M[AR \leftarrow R1]$$



Aug 20, 24

Logic Micro-Operations

It specifies binary operations stored in registers. These operations consider each bit of the register seperately and treat them as binary variable.

For eg. →

The exclusive OR micro-operations with the content of two registers R1 and R2 is symbolized by the statement

P : R1 ← R1 ⊕ R2.

It specifies a logic micro-operation to be executed on the individual bits of the registers provided that the control variable using method P = 1 as a numerical example. Assume that each register has 4 bits. R1 be 1 0 1 0 and the content of R2 be 1 1 0 0. The exclusive OR micro-operation gives the following computation.

as per X-OR,

```
1 0 1 0      R1
1 1 0 0      R2
0 1 1 0      new value of R1
```

The content of R1 after the execution of the micro-operation is equal to the bit by bit exclusive-OR operation on pairs of bits in R2 and previous value of R1.

Special symbols are used for the logic micro-operations OR, AND and NOT (compliment).

Symbols :→

$\vee \rightarrow$ OR

$\wedge \rightarrow$ AND

$\sim \rightarrow$ NOT

List of logic operations :

There are 16 logic operations that can be performed with 2 binary variables x and y. The functions are determined from the 16 binary combinations that can be assigned to F.

Truth Table of

X	Y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

$f_0 = 0$

$f_1 = A \wedge B$

$f_2 = A \wedge \bar{B}$

$f_3 = A$

$f_4 = \bar{A} \wedge B$

$f_5 = B$

$f_6 = A \oplus B$

$f_7 = A \vee B$

$f_8 = \overline{A \vee B}$

$f_9 = \overline{A \oplus B}$

$f_{10} = \bar{B}$

$f_{11} = A \vee \bar{B}$

$f_{12} = \bar{A}$

$f_{13} = \bar{A} \vee B$

$f_{14} = \overline{\bar{A} \wedge B}$

$f_{15} = 1$

Aug 22; 24 Gates

① AND gate
If both I/P are high, O/P is high

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

② OR gate
If one or more I/P are high, O/P is high

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

③ NOT gate
O/P is inverse of I/P

A	Y
0	1
1	0

④ NAND gate
high O/P if only one or both I/P is low.

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

⑤ NOR gate
high O/P if both I/P is low

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

⑥ X-OR
gives a high O/P if one of its I/P is high not both

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

high if I/P is different.
low if I/P is same.

(7) X-NOR
it gives low O/P if one of its I/P is high

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

high if I/P is same.
low if I/P is different.

Hardware Implementation

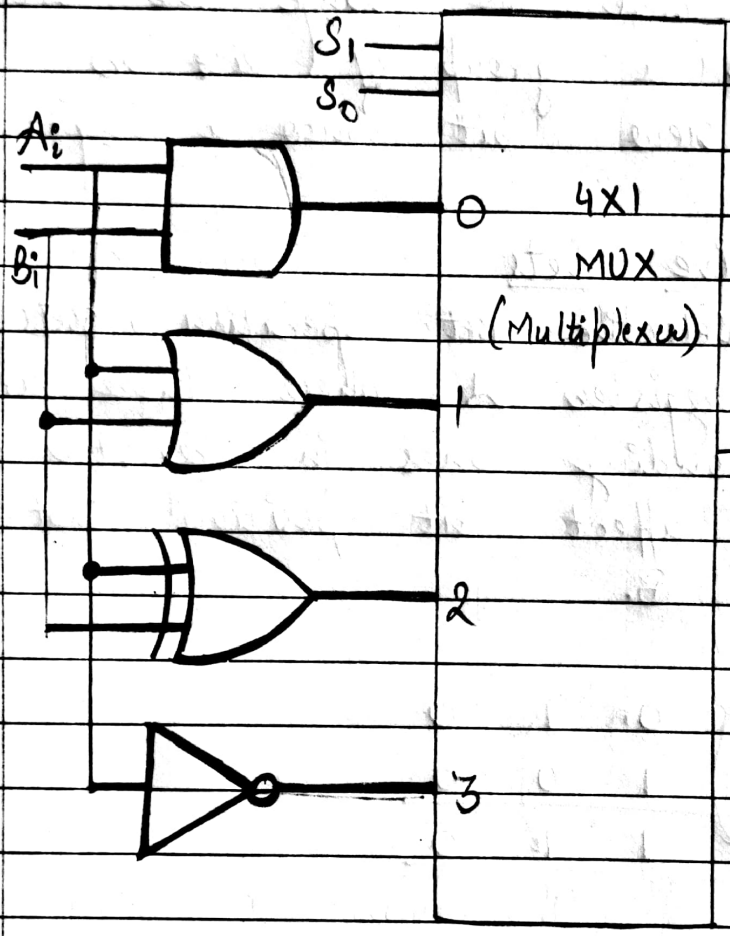
(1) The hardware implementation of logic micro-operations required that logic gates be inserted for each bit or pair of bits in the registers to perform the logic functions. (2) Although there are 16 logic micro-operations but mostly use only 4, i.e., AND, OR, X-OR, NOT

These operations are also used for manipulating the bits stored in a register.

(3) It consists of 4 gates and multiplexers each of the 4 logic micro-operations is generated through a gate that performs a required logic.

Aug 27, 24

(iv) The O/P of the gates are applied to the data I/P of the multiplexer. The two selection I/P's S1 and S0 choose one of the data I/P's of the multiplexer and direct its value to the O/P.



ONE STAGE OF LOGIC CIRCUIT

S_1	S_0	Output	Operation
0	0	$E = A \cap B$	AND
0	1	$E = A \vee B$	OR
1	0	$E = A \oplus B$	XOR
1	1	$E = \bar{A}$	(Complement) NOT

Logic - Microoperations Applications

① It is used for manipulating individual bit stored in a register.

② They can be used to change the bit value delete a group of bits or insert a new bit into a register.

for eg. → ① Selective Sets

The selective set operation sets to 1 in register A where there are corresponding ones in register B. It does not affect bit positions that have zeroes in B.

A → 1 0 1 0
B → 1 1 0 0
1 1 1 0

② Selective Complement

The selective complement operation compliment the bit in A where there are corresponding ones in B. It doesn't affect bit position that has zeroes in B.

complement
A → 1 0 1 0
B → 1 1 0 0
0 1 1 0

⊙ Selective Clear

Shift Micro-operations

1. Shift micro-operations are used for serial transfer of data.
2. The content of register can be shifted to the left or to the right.
3. During a shift left operation, the serial I/P transfer a bit into the right-most position.
4. During a shift right operation, the serial I/P transfer a bit into the left-most position.
5. There are three types of shift operations:-
   a) logic
   b) circular
   c) arithematic

Logical Shift Micro-operation

It transfer zero (0) through the serial I/P.
A symbol 'shl' for logical shift left and 'shr' for logical shift right.

The micro-operation that specifies a 1 bit shift to the left of the content of register R and 1 bit shift to the right of the content of register R as shown below.

The bit transfered to the end position through the serial I/P is assumed to be (0) zero during a logical shift.

Date : / /
Page :

Circular Shift Micro-operation

It is also known as Rotate Operation.

It circulates the bit of a register around the 2 ends without loss of the information.

This is accomplished by connecting a serial I/P of the shift register to its serial O/P. We will use the symbols Cil for the shift left and Cir for the circulate shift right.

Arithematic Shift Micro-operation

It is a micro-operation that shifts the signed binary number to the left or right.

An arithematic shift left multiplies a signed binary number by 2 and arithematic shift right divides the number by 2.

Arithematic shift must leave the signed bit unchanged because the sign of a no. remains the same when it is multiplied or divided by 2.

Advantages → 1. Productive Information Controlled.
2. Upgraded number crunching activity
3. Bitwise Activity
4. Utilization of memory

Applications → 1. Math activities
2. Design handling

XTRA EDGE .....
Date : / /
Page :

UNIT- 1 BASIC COMPUTER ORGANIZATION

* Instruction Code:

A computer instruction is a binary code that determines micro-operation in a sequence to computer.

Each computer has its specific group of instruction. They can be categorized into 2 elements:

① Operation code (OP code)
② Address code

1. OP code → Specific instruction operation perform on specific instruction.

2. Address code → An address means which register can be used for this operation.

For eg. → If there is 12 bits of memory that are required to define the address mode. Therefore, the 15th bit of the instruction determine the address mode.

Therefore, the instruction format includes 12 bits of address and one bit for the addressing mode and 3 bit for the address.

0 - 12 bits
12 → instructions
1 → address mode
16th bit → 2 bits ← OP code.

1	OP code	Memory Address

→ There are 3 parts of instruction format →

① Addressing Mode
(a) Direct Addressing → for address of the operand
(b) Indirect Addressing → It uses for the pointer of operand.

(pointer → use to display & known the address)

② Effective Address
It defines the address that can be executed as a target address for a branch type instruction or you can say that the address that can be used directly to create an operant for a computation type instruction without creating any changes.

XTRA EDGE

Date : / /
Page :

OP code (Operation code)

An OP code is a collection of bits that represent the basic operations including add, subtract, multiply, complement and shift. The total no. of operations provided through the computer determins the no. of bits needed for the OP code.

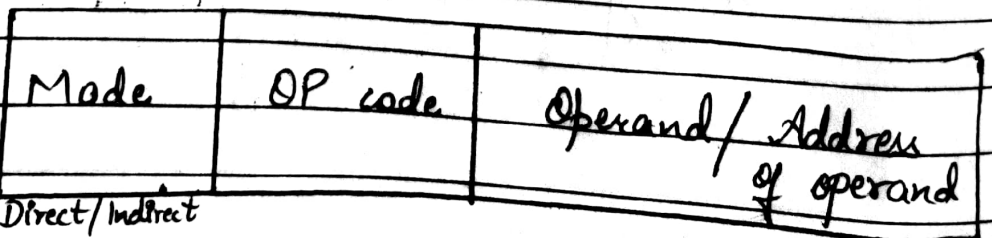
(if n no. of bits = n no. of operations performed)

Address

The address is represented as the location where a specific instruction is constructed in the memory. The address bits of an instruction is used as an operand and not as an address. In such methods, the instruction has an immediate operand.

If the second part has an address, the instruction is referred to have a direct address.

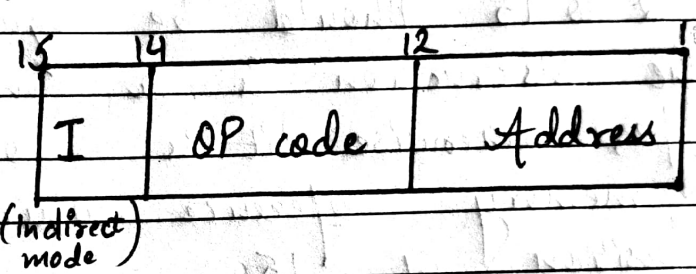
There is another second part including the address of an operand. This is referred to as an indirect address. Only one bit can signify if the direct or indirect address is executed, i.e., 0 and 1.



A basic computer has 3 instruction code formats which are -

1. Memory - Reference instruction
2. Register - Reference Instruction
3. Input - Output instruction

1. Memory Reference Instruction



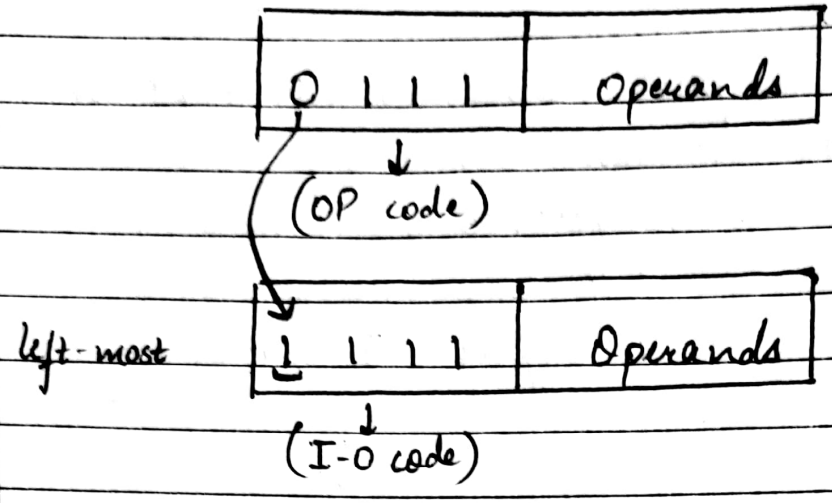
In memory reference instruction, 12 bits of memory is used to specify an address and 1 bit to specify the addressing mode.

2. Register Reference Instruction

They are represented by the OP code which is used 4 bits in a diagram.

3. Input Output Instruction

Just like the register reference instruction an I-O instruction does not need an any reference to into a memory and it is recognised by the operation code with a 1 in the left-most bit of the instruct



The 3-operation code bits in position 12 to 15 should be equal to 111 otherwise the instruction is a memory reference type, otherwise it is considered as a memory reference which is on the position of 15.

Computer Instruction Timing and Control

The timing for all registers in the basic computer is controlled by a master clock generator. The clock pulses are applied to all flip-flops and registers in the system.

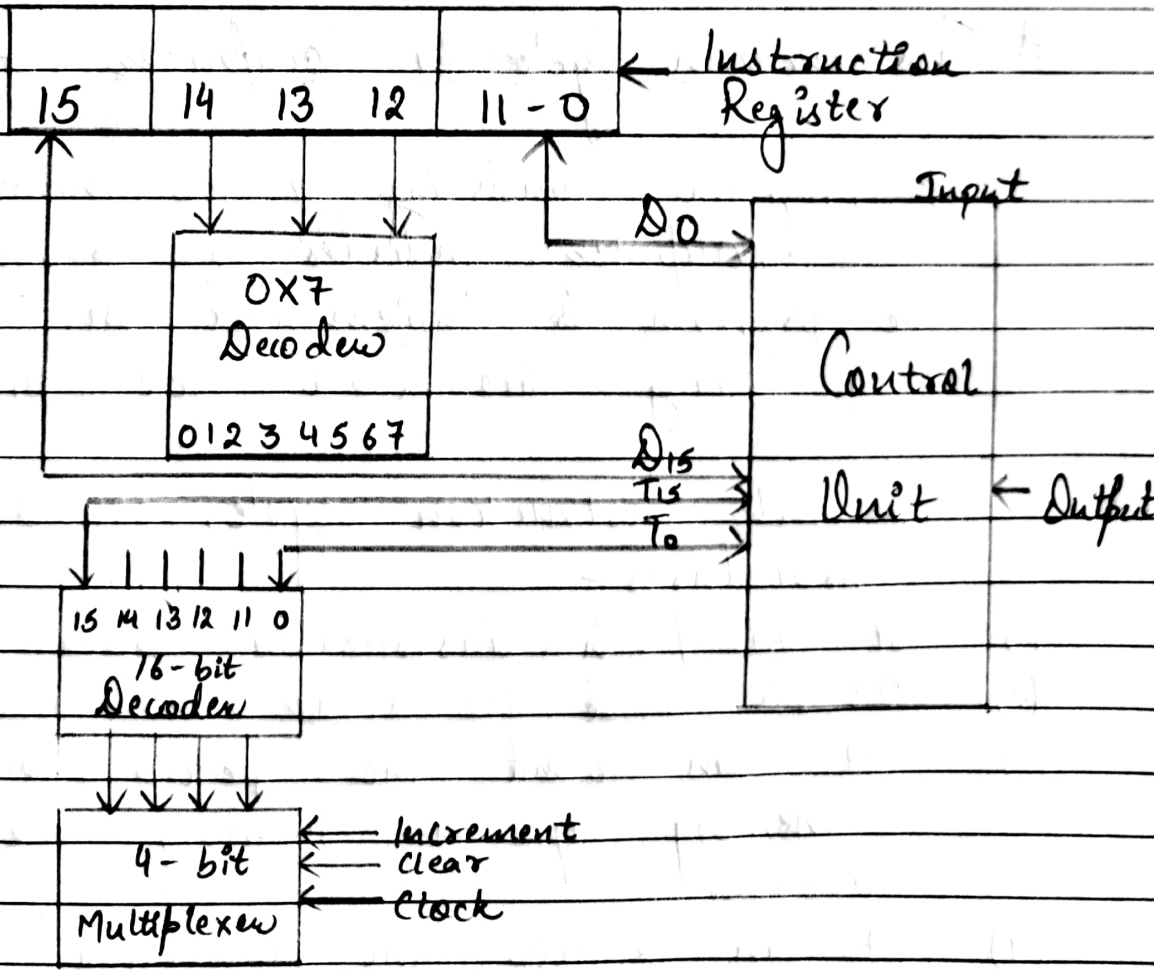
The control signals are generated in the control unit and provides control input for the multiplexers in the common data bus.

Control input in the processor registers and micro-operators for the manipulations.

There are 2 types of control organisation →
① Hard wired control
② Micro-programmed control

① Hard Wired Control
It is implemented with gates and other digital circuit (like multiplexers, decoders).
It can be used to provide a fast mode of operations.

② Micro-programmed Control
The control instruction is taught in a memory is programmed to initiate the required sequence of micro-operations.



Control Unit of basic computer

Date : / /
Page :

Clr → Clr input is active when positive clock transition happens. The sequence of timing signals $T_0, T_1, T_2, \cdots$ shown in the diagram. It can represent the relationship b/w the timing signals and its corresponding positive clock transition.

for eg. → Consider the case incremented instruction timing signals are T_0, T_1, T_2, T_3, T_4 in the sequence. At time of T_4, cleared section is zero and it is active after the incrementing will complete.

Sept 9:24

Instruction Cycle in Computer Architecture

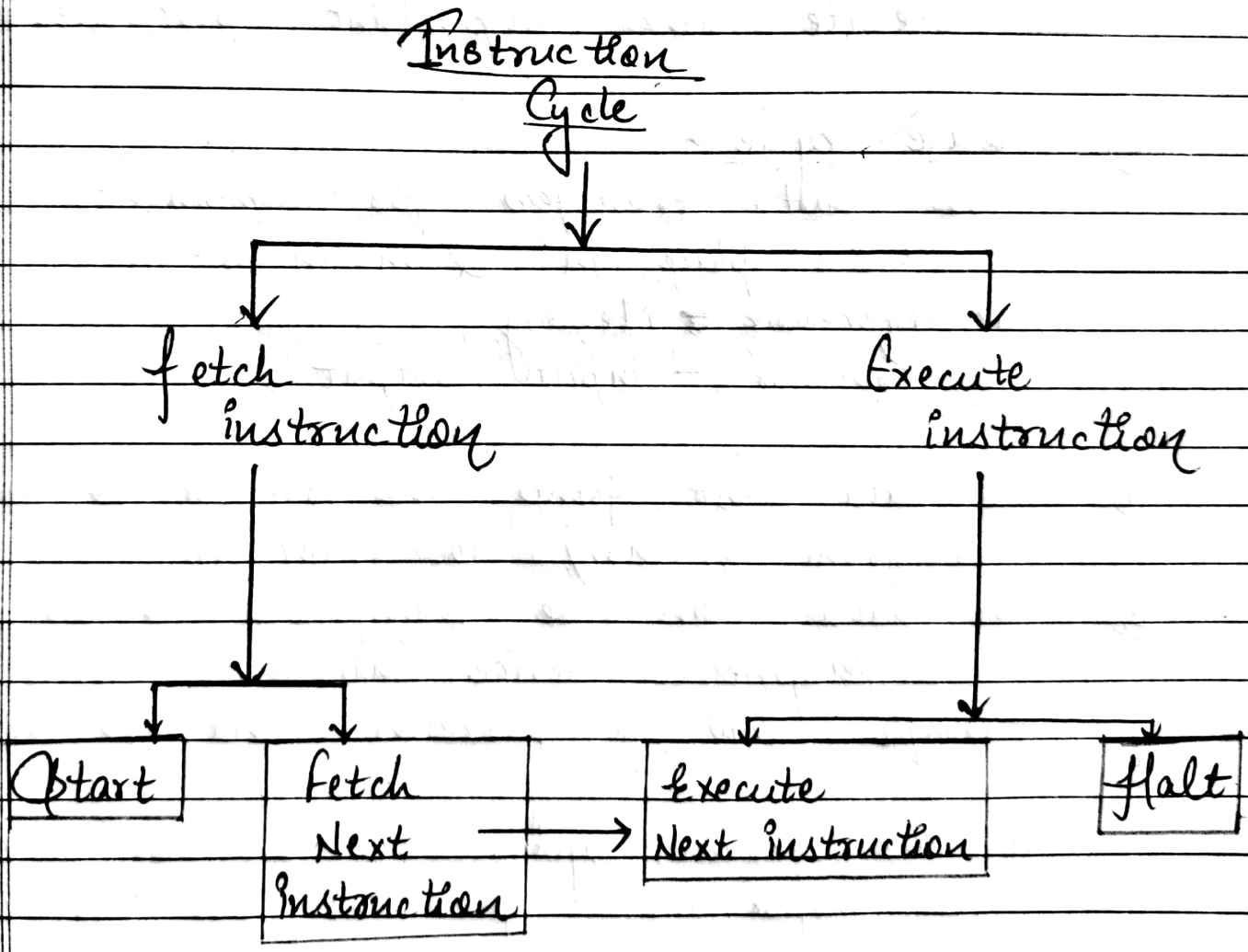
A program consisting of the memory unit of the computer includes a series of instructions. The program is implemented on the computer by going through a cycle for each instruction.

In each instruction cycle, follow the different procedures -

1. It can fetch instruction from memory.
2. It is used to decode the instructions.
3. It can read the effective address from memory if the memory has indirect address.
4. It can execute the instruction (procedure)

After the following 4 procedures are done, the control switch back to the first step and repeat the similar process for the next instructions.
Therefore, the cycle continuous until a halt condition is met.

The figure shows the faces contain in the construction cycle.



① **Fetch Cycle →**

The instructions are stay in program counter which are going to execute.

The processor fetches the instruction from the memory, i.e., pointed by the PC (program counter)

Next, the PC is incremented to display the address of the next instruction. The processor reads the instructions and execute the important procedure.

② **Execute cycle →**

The data transfer for implementation takes place in 2 methods -

① Processor ✱ Memory
② Processor – Input / Output

① The data sent from the processor to memory or from memory to processor.
② The data can be transferred to or from a peripheral device (hardware device) by transfer b/w a processor and I/O device.

In the execute cycle, the processor implement the important operations on the information. These 2 methods associate and complete the execute cycle.

Date : / /
Page :

State Diagram for instruction cycle →

explain → operand → operation

A. (1) Instruction Address Calculation →
The next instruction address is computed here. A permanent ~~numb~~ number is inserted to the address of the earlier instruction.

Input - Output
no. of bits n 2n (nm)

(2) Instruction Fetch →
The instruction is read from its specific memory location to the processor.

(3) Instruction Operation Decoding →
The instruction is interpretted and the type of operations to be implementing and the operand(s) to be used are decided.

B. (4) Operand Address Calculation →
The address of the operand is evaluated if it has a reference memory through the input - output.

(5) Operand Fetch →
The operand is read from the memory or from I/O.

(6) Data Operation →
The actual operation is executed or last store operands. It can store the result in

the memory or transfer it to I/O.

Sept 10;24

Memory Reference Instruction

There are seven memory reference instructions. The Decode Output (D_i) for $i = 0, 1, 2, 3, 4, 5$ and 6 from the operation decoder that belongs to each instruction is including in the table.

The effective address of the instruction is in the address register AR and was placed there during timing signal T_2 when $i = 0$ or during timing signal T_3 when $i = 1$.

The execution of the memory reference instructions start with timing signal T_4.

The symbolic description of each instruction is specified in the table in terms of register transfer notation.

Memory Reference Instruction Table

Symbol	Instruction/Operation	Decoder	Symbolic Description
AND		D_0	$AC \leftarrow AC \wedge M[AR]$
ADD		D_1	$AC \leftarrow AC + M[AR]$, $E = cout$
LDA		D_2	$AC \leftarrow M[AR]$
STA		D_3	$M[AR] \leftarrow AC$
BUN		D_4	$PC \leftarrow AR$
BSA		D_5	$M[AR] \leftarrow PC$, $PC \leftarrow AR + 1$
ISZ		D_6	if $M[AR] + 1 = 0$ then $PC \leftarrow PC + 1$

AC → Accumulator

(1) AND to AC

This is an instruction that performs the AND logic operation. The result of the operation is transferred to AC. The micro-operation that execute this instruction are- $D_0 T_4$.

(2) ADD to AC

This instruction adds the content of the memory specified by the effective address to the value of AC.

eg.→ Accumulators are registers.

The sum is transferred into AC and the O/P carry out is transferred to the E accumulator, E = extended, i.e., extended accumulator (eg→ flip flop).

The micro-operations needed to execute these instructions are $D_1 T_4$, $D_1 T_3$.

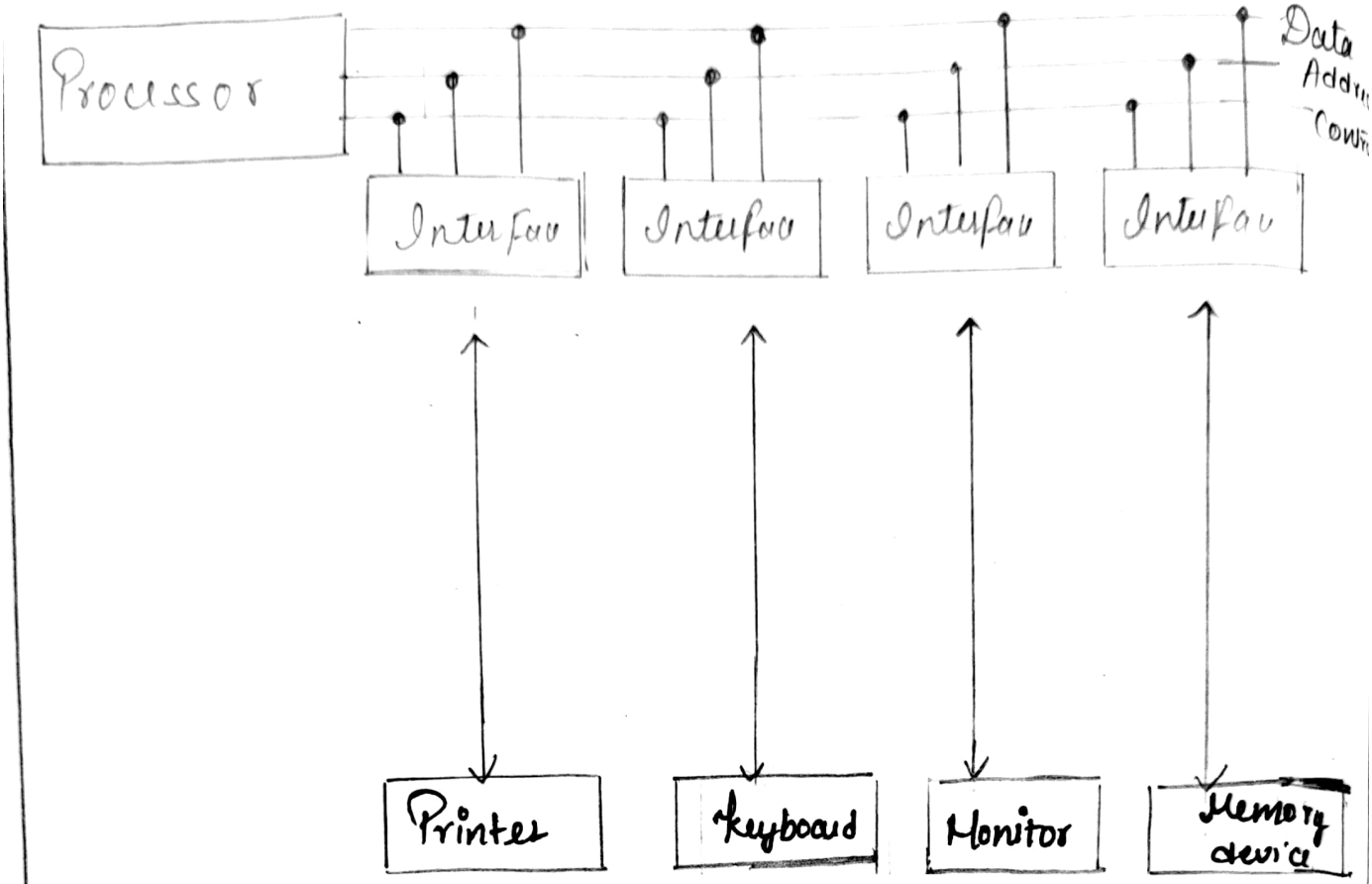
Table

(3) LDA (load to AC)

This instruction transfer the memory word specified by the effective address to AC.

The micro-operations needed to execute are $D_2 T_4$, $D_2 T_5$.

Input/ output interface:



I/O Interface diagram....

The I/O Interface implement a method by which data is transfered b/w Internal Storage and external I/O devices.

All the devices are connected to a Computer that require a special Communication Connection for Interfacing with the CPU.

I/O bus is the route used for Peripheral devices to interact with the CPU or hardware device

Computer and processor. The fig shows the typical connection of the I/O to I/O device.

The input output bus is linked to all devices and the device address with the interface.

An interface receive four commands.

(1) Control :→ A Command Control is given to activate the device and inform its next task. Each device has a sequence of control commands.

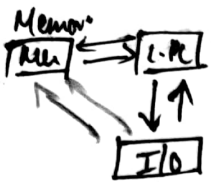
(2) Status :→ It can check or test multiple test conditions. On the interface and the device.

(3) Data o/p → It is used to Create the Interface counter to the Command by sending data from the bus to one of its Register.

(4) Data i/p →

This Command is opp to the data o/p. On this, the interface gets the data from Hardware device and copy it to specific Register.

D.T b/w the CPU & I/O devices may be handled in variety of ways.



There are 3 modes of transfer :

(1) Programmed I/O : Here the data is transferred by an Instruction written in the CPU as an intermediate path. programmed and it can be happened From Register and Memory also.

(2.) Interrupt-Initiated I/O :- By using Interrupt facility a Special Command to inform the interface to issue an interrupt request signal. when data is available Fromm many devices. The CPU can proceed For any programme. when it is determine that the device is ready For data transfer that Initiate request Signal to the Computer Fm the CPU stop the task that was already performing.

3 DMA → The Data transfer b/w a Fast Storage Media Such as Magnetic disk and Memory until the Speed of CPU is limited avg.

Thus, we can allow the device, that directly Communicate with each other using bus

DMA:

DMA: DMA is a direct memory access act as a Station Master that transfer the data with minimum intervention of the Processor.

Hardware device:

The hardware device used for direct memory Access is called DMA Controller. DMAC is a Control unit part of in/output device interface Circuit. which can blocks of data b/w in/output devices and main-memory without processor.

DMA Controller Architecture:-

DMAC provides an interface b/w the bus and the in/output devices. Although it transfer the data without the intervention of Processor but, it is Controlled by the processor. The processor initiates the DMA Controller by sending the starting address, number of words and direction of data i.e From in/output devices to the main memory.

More than one external device can be connected to the DMA Controller.

Working of DMA Controller :-

It has to share the bus with the processor to make the data transferred the device that holds the bus at the given time is called the "Bus Master".

When a transfer from I/O device to memory or vice-versa. has to be made. In this situation the processor stop the execution of current program of the DMAC is free, it requests the control of the Bus from the processor by raising the bus request signal. Processor grands the Bus request signal by raising the grant signal

Now the Bus Master.

The processor initiates the address and the no. of bits to be transferred after the task is complete. The processor resumes the execution of the program after retreiving instruction from the DMAC Controller.

3 Modes of DMA Transfer :-

1.) Burst mode :- In this DMA handover the buses to CPU after the completion of the task (Data transfer)

2.) Cycle Stealing mode :- In this DMA give control of buses to CPU after transfer of every byte.

3.) Transparent mode :- Here DMA transfer data only when CPU is executing the instruction, which does not require the use of buses.

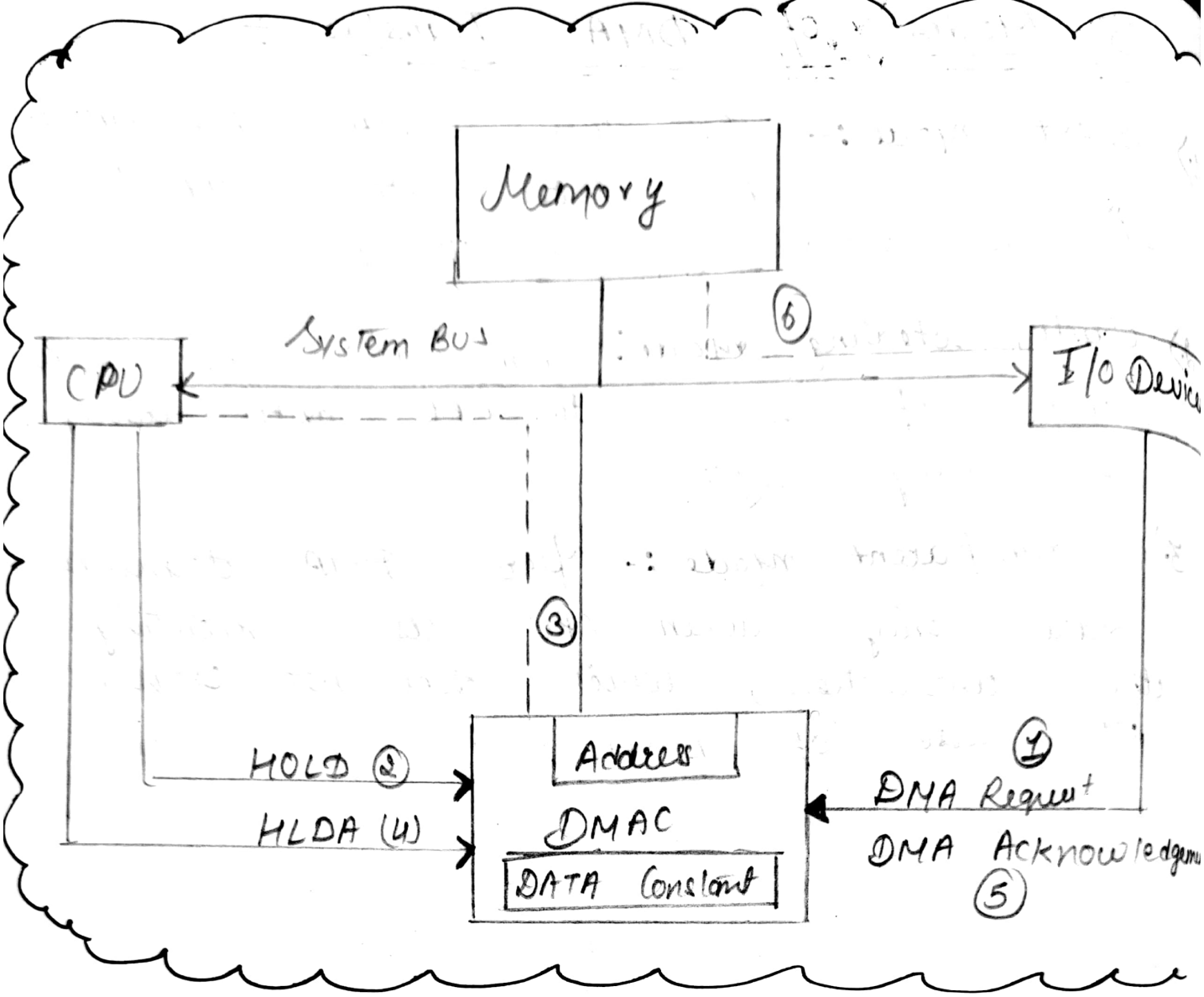


Fig - DMA Controller

Control unit : way to implement CU are —

(I) Hard wired (II) Micro-programmed

→ CPU Consist unit →

(I) CU (III) MU → Bahr.

(II) ALU (IV) I/p unit → (V) O/p unit

CU → It is a virtual component of a Computer CPU, responsible for co-ordinating and controlling the activities of various Hardware components.

→ The CU receives instructions in the form of a binary code from the memory. It decode these inst.. to determine an specific operation that need to be performed. Then it generate control signals that are sent to diff part of CPU such as ALU, Reg etc to carry out operation

Computer Registers :

Registers are type of computer memory used to quickly accept store and transfer data by the CPU.

Registers are used by the CPU known as Processor Registers.

A processor register hold on instruction storage address or Data. The computer needs to manipulate data by the registers and the memory location calculate the address for next instruction of next instruction after the execution of current instruction is completed.

(1) Data Register :

Symbol : DR

No. of bits : 16

Function - hold memory operations

(2) Address Register :

Symbol : AR

No. of bits : 12

Function: hold the address for the memory

(3) Accumulator :

Symbol : AC

no. of bits : 16

FN : Processor Register

(4i) Instruction Register

Symbol : IR

no. of bits : 16 Bits

Fn : Hold instruction code

(5.) Program Counter.

Symbol : PC

no. of bits : 12 bits

Fn — Holds the address of next instruction

6.) Temporary Register.

Symbol : TR

no. of bits : 16

Fn : Holds Temporary Data

(7.) Input Register

Symbol : INPR

no. of bits : 8 bits

Fn : Carries input Characters

(8.) Output Register

Symbol : OUTR.

No. of bits : 8

Fn — Carries O/p characters

Register & Memory Configuration

(1) The Memory Unit has a Capacity 4096 words and each word Contain 16 bits.

(2.) The DR Contain 16 bits which holds the operand read from the memory location.

(3.) The Memory Address Register (MAR) Holds the Address for the Memory locations.

(4) The SC means Status Control or Status Condition.

Basic operational Concept :

(1) This instructions are stored in a memory and executed to Process Data which are loaded in the Computer Memory through the input Device.

(2.) After processing the Data the result is either stored in the memory result for Further reference or Send to the outside world through some o/p device.

(3) To perform the execution of an instructions including Arithmetic and Processor Contains nos of Registers used for Temporary storage.

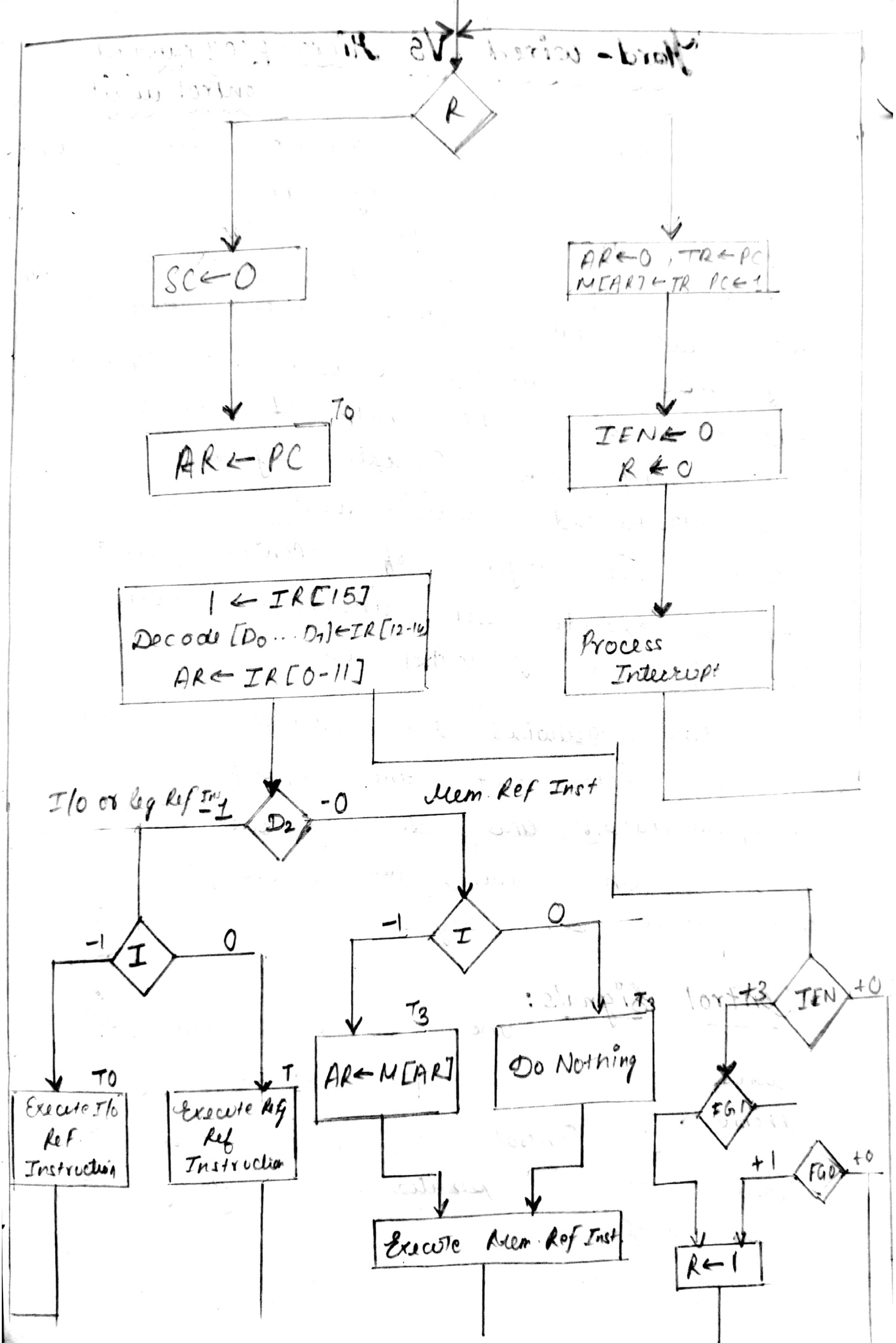
(4.) The Program Counter is one of the most Critical Register in C.P.U. which keeps track on the next instructions being executed. (5) The IR register generate the timing Signal that Control timmig the various processing elements involved in a executing the instruction.

(6) The two Registers MAR & MDR used to handle the data transfer b/w Main-Memory and the Processors.

Complete Description of Computer

The Complete Computer description of a Computer includes :-

It includes both Instruction cycle as well as the Interrupt Cycle. The interrupt cycle because there may arise a case of input output operations any-time during normal operations. Instruction cycle because in the absence of interrupt the CPU is always busy with Stored programme instruction & Flip-Flop are used as a Condition to determine the type of operation. when R = 0, the instruction cycle Continuous and when R=1; interrupt cycle Continuous. So, an interrupt is signaled by IEN (i.e) Instruction Enabled number and 'R' is just to make it ensure that another interrupt does not get Entertain while Processing of an interrupt.


R
SC ← 0
AR ← 0 , TR ← PC
M[AR] ← TR PC ← 1
T0
AR ← PC
IEN ← 0
R ← 0
I ← IR[15]
Decode [D0 ... D7] ← IR[12-14]
AR ← IR[0-11]
Process Interrupt
I/O or Reg Ref Inst
D2
-1
-0
Mem Ref Inst
I
-1
0
I
-1
0
T0
Execute I/O Ref Instruction
T
Execute Reg Ref Instruction
T3
AR ← M[AR]
Do Nothing
T3
Execute Mem Ref Inst
T3
IEN
+3
+0
FGI
FGO
+1
+0
R ← 1

Control Unit generate the control signals

Hard-wired Vs Micro-Programmed Control unit

In a System or a Computer most of the tasks are controlled by CPU (Central Processing unit) which is the main Component of a Computer. The CPU usually has two main parts - Control unit (CU) and ALU The 'CU' is used to Synchronize the task. with the help of Sending the timing and control signals. AND Hardwired Control units can be. Called two types of control unit. ALU can execute an instruction with the help of these 2 control units.

In the Hardwired the execution of operations is much faster but the modification implementation and decoding are difficult.

In Contrast, In Micro-programmed these tasks are easy.

- **Control Signals:** The Control Signals are used to know about various types of theme ① Control Signals are used to know what operations is going to be Performed.

(2) To know about the sequence of operations that are performed.

(3) To know about the timing at which an operation must be executed and many other types of things.

Hardwired Control Unit:

With the help of generating Control Signals Hardwired Control Unit is able to execute the instructions at a correct time and proper sequence.

The Control-Signals are generated with the help of circuit and state counter.

The instruction register is a type of a processor register used to contain an instruction going to be executed. The IR generate an op-code of instruction bits and addressing bits.

The above generated op-Code all recieved in the field of an instruction decoder. The instruction decoder interprete the Operations & instructions.

Now on the basis of addressing mode. of instructions and operations which exist in the instruction Register.

The ~~IR~~ Decoder sets the ~~Co~~ponding

instruction signals. INS1, INS2 ....

Steps are used on each instruction are ① Instruction fetch, ② Decode, ③ Operand ④ fetch, ⑤ ALU and ⑥ location.

→ The information about the current step of an instruction must be known by the control unit.

Now, the step counter is implemented which is used to contain the signals from T1, T2 ..... T5 on the basis of the step one of the signal of step counter will be set from T1 to T5 to 1.

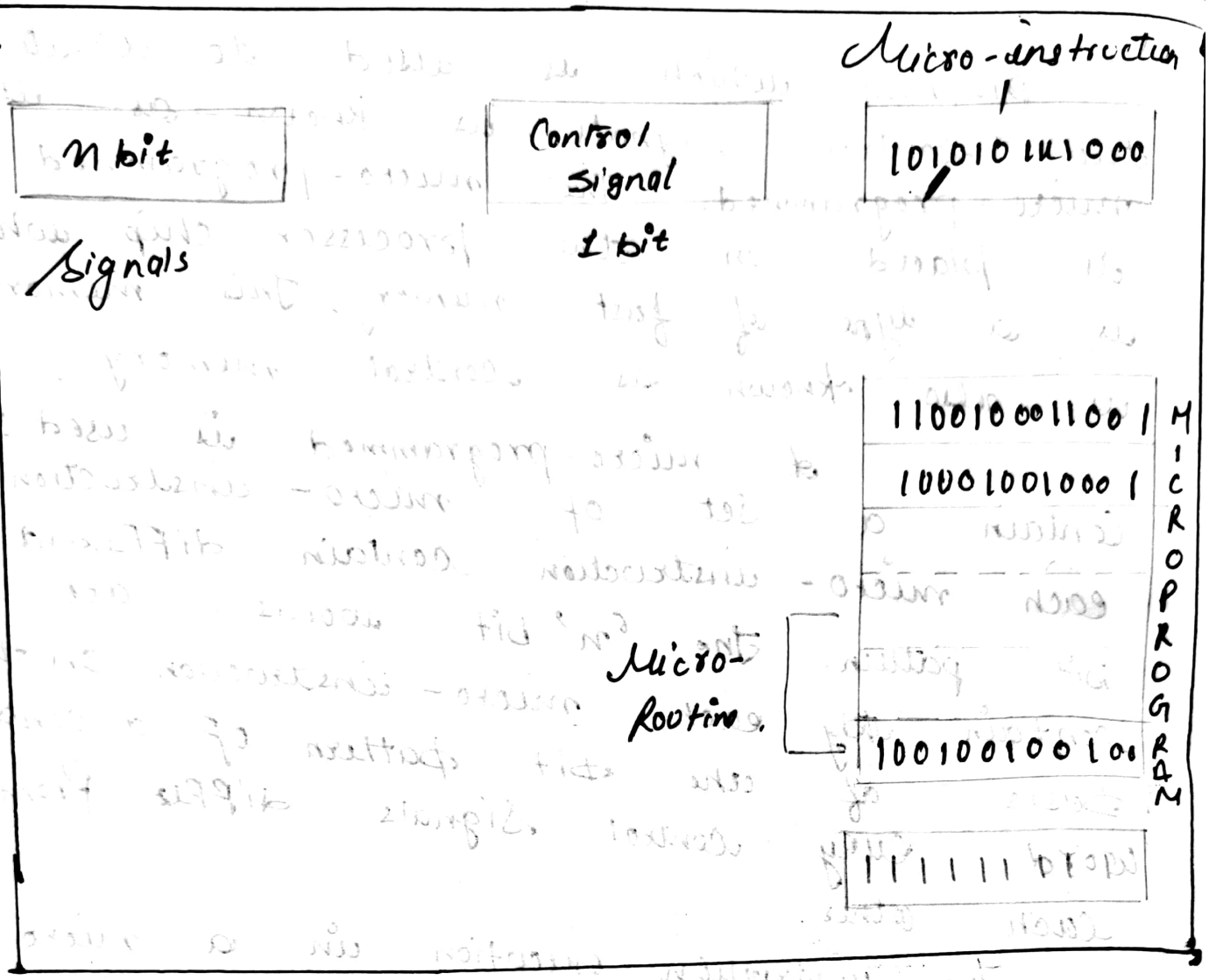
Micro-programmed Control unit:

A micro-programmed control unit can be described as a simple logic circuit we can use it in 2 ways. i.e it is able to execute each instruction. with the help of generating control signals and it is also able to do sequencing through micro-instruction. It will generate the control signals. with the help of programme. at the time of evaluation This approach was very famous.

The program which is used to create the control signals is known as the micro-programmed. The micro-programmed is placed on the processor chip which is a type of fast memory. This memory is also known as control memory.

A micro-programmed is used to contain a set of micro-instruction. each micro-instruction contain different bit pattern. The 'n' bit words are contain where each micro-instruction on the basis of the bit pattern of a control word. every control signals differ from each other.

The instruction execution in a micro-programmed control unit is also performed in each steps so for each steps the micro-programmed contains a micro-instruction. If we want to execute a particular instruction. These process is known as micro-routine.



Micro-programmed Control Unit Organization:

(1.) **Instruction fetch:** It is the first step in this step, the instruction fetch from the IR with the help of micro-instruction 'AR'

(2.) **Decoder:** It is 2nd step in this step the instruction is obtain from the IR will be decoded with the help of a micro-instruction address generator.

(3) **Increment:** It is the 3rd step. The

Value of the micro-programmed Counter will be increased so that it can execute the next micro-routine.

(4) 'End' bit is the fourth bit step. Micro-routine contain a bit known as end bit. When instruction will be successfully Complete. The end bit is set to '1'.

(5.) The last step the micro-programmed address generator will again go back to step '1'. So that it can fetch a new-instruction.

Difference:

HARDWIRED CONTROL UNIT	Micro-programmed Control Unit
(1) With the help of hardware circuit we can implement only the hardwired control unit	(1.) In this approach we use programming
(2.) HCU used control circuit and generate the control signals which are required by the processor	2.) It is very easy to modify
(3.) It is very difficult to modify.	(3.) It used micro-instructions and generate the control signals required memory for this.
(4) In the form of logic gates everything has to be realized in the HCU that's why its very Costly	(4) less Cost because the control unit Only requires the Micro-Operations

5. The Complex I/s cannot be handled by Hardwired Control Unit bcz when we design a ckt it will become Complex	It is able to handle the Complex I/s
(6.) limited no. of I/s	Many I/s
(7.) used for types of computer that implement RISC (Reduce I/s Set Computer)	used for CISC (Complex I/s Set Computer)
(8.) It is Faster	It is slow.

Unit-2

Part II

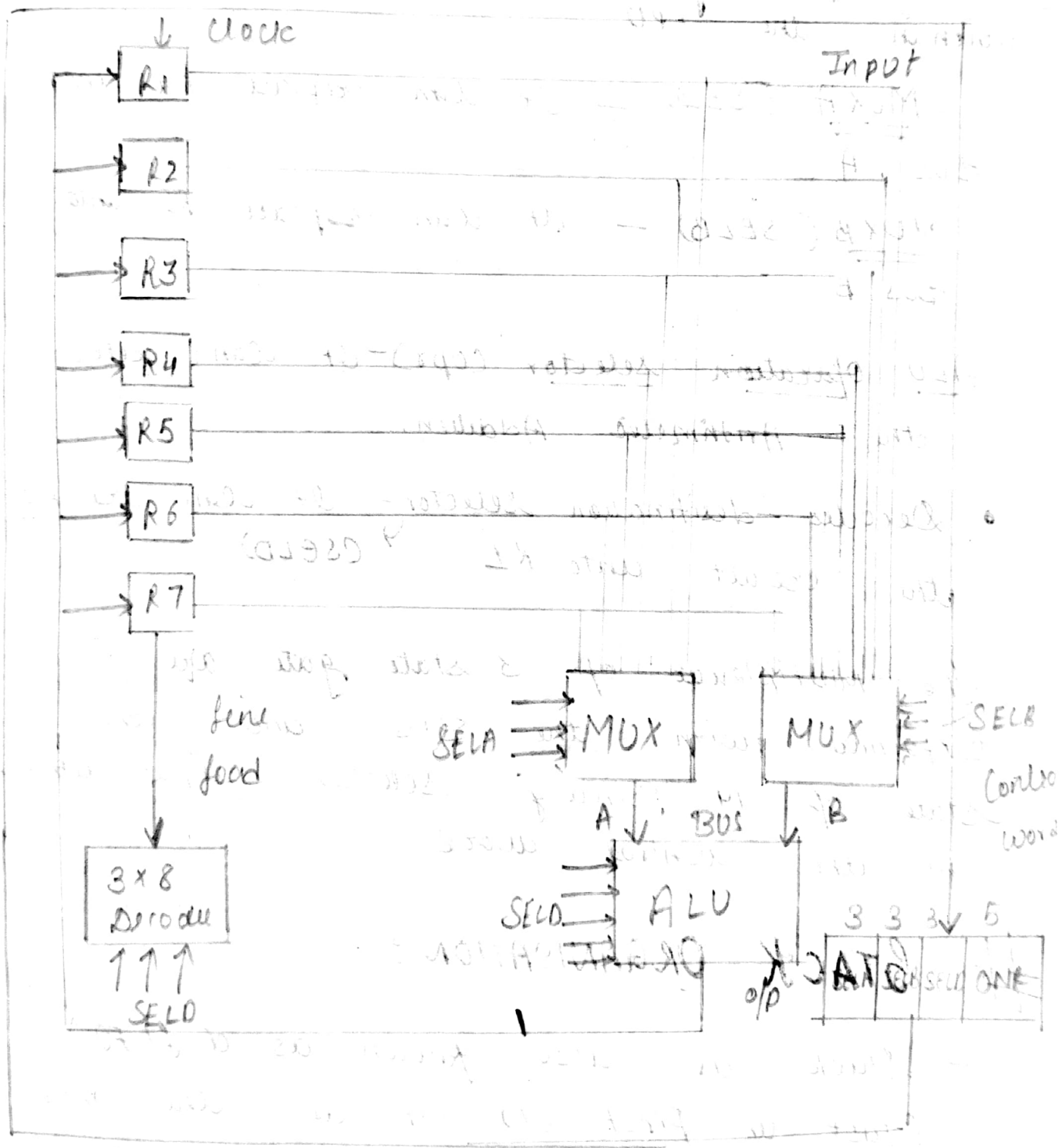
CENTRAL PROCESSING UNIT

General Register Organization:

A set of flip-flops forms a Register. A register is a unique high speed storage area in the CPU. They include combinational ckt that implement Data Processing. The information is always defined in a Register before processing. The Register speed up the implementation of the Program.

There are 2 important function of the Register in the CPU operation.

(1.) It can spot a temporary storage location for data. It means it directly Implemented programs to have fast access. It can save the status of the CPU and data about the directly Implemented programme.



The CPU bus system is manage by the Control unit.

The control unit explicit the data flow through the ALU. By choosing the function of the ALU and component of the system.

Consider R1 ← R2 + R3

The following are the function implements within the CPU.

- MUX A (SELA) – It can replace R2 into bus A.
- MUX B (SELB) – It can replace R3 into bus B
- ALU operation selector (OPR) – It can select the Arithmetic Addition.
- Decoder destination selector – It can transfer the result into R1 (SELD)

The Multiplexer of 3 state gate are performed with the Bus, and the state of 14 Binary selection input determine the control word.

STACK ORGANISATION:

– Stack is also known as a LIFO (Last in First out). It is the most important feature in CPU. It saves

data such that the element store last is retrieved fast. A stack is a memory unit with an address register.

This Register influence the address for the stack which is known a Stack Pointer (SP)

The Stack Pointer continually influences the address of the element that is located at the top of the Stack..

- It can insert an element into or delete an element from the Stack. The insertation operation is known as Push - operation. and the deletion operation is known as stop - Operation. In a Computer Stack. This Operations are simulated by incrementing or decrementing the SP Register.

Register Stack:

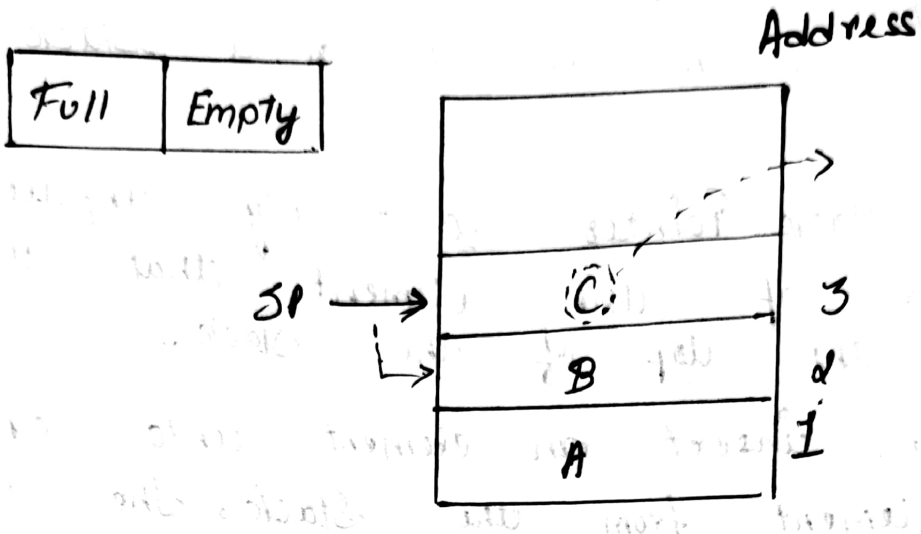
The stack can be arranged as a set of memory words or Register.

Consider a 64 words Register Stack arranged. The SP Register includes a binary number which is the address of the element present at the top of the stack.

The 3 Elements A, B and C are located in the Stack. The element 'C' is at the top of the Stack i.e 3

The top element is opt from the Stack

through reading memory word at address '3' and decrementing the stack pointer by '1'.



Then B is at the top of the stack and the SP holds the address of B i.e 2 It can insert a new word. The stack is pushed by incrementing the stack Pointer by 1 and inserting a word. in that incremented location.

The stack pointer includes 6 bits. because $2^6 = 64$ and the SP cannot exceed. 63. After all if 63 is simply - incremented by 1. Therefore the result is zero.

SP holds only the 6 L.S.B.

:, when the stack is full the one bit register FULL is said to 1

If the stack is null then the One bit Register empty is said to zero.

The Data Register (DR) Holds the binary information which is composed into or read out of the Stack.

First, the SP is set to 0, EMTY is set to 1, and Full is set to 0. Now, as the Stack is not Full (Full=0), a new element is inserted using the push operation.

The Push operation is executed as Follows -

• SP ← SP + 1	It can increment stack pointer.
• M[SP] ← DR	It can write element on top of the Stack.
• If (SP=0) then (FULL ← 1)	Check if Stack is Full.
• EMTY ← 0	Mark the Stack not empty.

The Stack pointer is incremented by 1 and the address of the next highest word is saved in the SP.

The word from DR is inserted into the Stack using memory write operation.

The first element is saved at address 1 and the final element is saved at address 0.

If the SP is at 0, then the Stack is 'Full' is set to 1. This is the condition

when the SP was in location 63 and after incrementing SP, the final element is saved at address 0. During an element is saved at address 0, there are no more empty registers in the stack. The stack is Full and the 'EMPTY is set to 0.

A new element is deleted from the stack if the stack is not empty (if EMPTY=0). The Pop operations includes the following sequence of micro-operations:

• DR ← M[SP]	→ It can read an element from the top of the stack
• SP ← SP − 1	→ It can determine the stack pointer
• IF (SP = 0) then (EMTY ← 1)	— Check if stack is empty
• FULL ← 0	— Mark the stack not Full

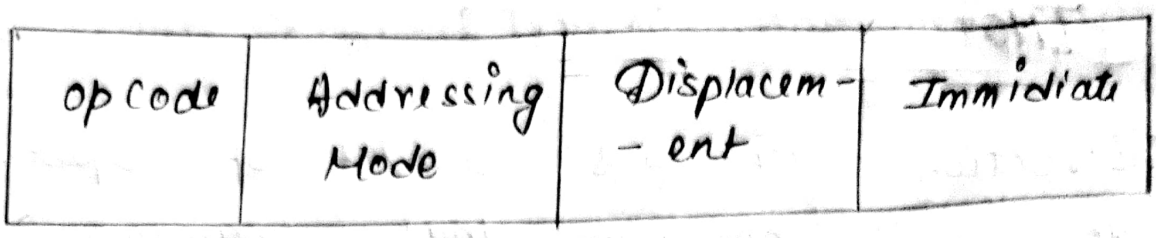
The top element from the stack is read and transfer to DR. and thus the stack pointer is decremented. If the stack pointer reaches 0, then the stack is empty and 'EMTY' is set to 1. This is the condition when the element in location 1 is read out and the SP is decremented by 1.

INSTRUCTION FORMAT :

1) Instruction includes a set of operation code and operands, that manages with the operation codes.

- Instruction Format supports the design of bits in an instruction.
- It contains fields including op code, operand and addressing mode.
- The instruction length is generally observed in multiple of the character length, which is of 8 bit.
- When the instruction length is permanent several bits are assigned to op-code, operands and addressing mode.
- The function of allocating bits considered the following elements :-
  - (i) No. of addressing mode.
  - (ii) Number of operand.
  - (iii) Number of CPU registers.
  - (iv) Number of register sets.
  - (v) Number of address lines.

- The journal architecture of 32 bits instruction format, micro-processor uncod 4-field :-



- In the addressing mode field instruction needs only 1 byte.
- The field that directly follow the addressing mode is the displacement field.
- If an effective address for a memory operand is computed using the displacement value then it uses 1 byte or more byte.
- If an operand is an immediate value then it is located in the immediate Field and uses 1 byte.

ADDRESSING MODE:

An instruction contain an operation Field and address Field and the mode Field.

→ The operation Field indicates what operation is to be performed & for example; Addition Subtraction etc.

→ The mode field indicates how the memory address of the operand which is supplied in operation is determined

Types of Addressing Mode:

To understand the various types of Addressing Mode It is important to understand how the Computer deals with instruction.

The instruction cycle of a Computer goes through the following 3 Phases :-

(1.) Fetch the instruction from the memory

(2.) Decode the instruction.

(3.) Execute the instruction.

TYPES:

(1.) Implied mode :

— In this mode, the instruction contains an indirect definition of the operand.

For eg — CMA (Complement Accumulator) operant means complement there.

2. Immediate A. Mode:

In this mode the instruction contains both op code and the operand. It can be said that an instruction that uses immediate Addressing mode contain an operand field in place of an address field

eg → ADD the instructions which is operation and number 10, which is operand.

3. Register Mode:- In this Mode the instruction specifies a Register. This Register store the operand and example of Register Mode is AC = AC + [R]

This will add the operands stored at the Register R. to the operand store in the Accumulator.

:) Register Indirect Mode:-

In this Mode, the instruction specify a register. This register store the effective address of an operand.

eg → AC = AC + [[R]]

Here the content which resides in the Memory location specified by the register are will be added to the content of the Accumulator.

(5.) Direct Addressing Mode: In this Mode, the instruction specifies the address. This address is the address of an operand.

For eg → AC ← AC + [X], this will add the operand store at address X with the operand store in the accumulator.

- This is also known as Absolute A. Mode.

(6.) Indirect Addressing Mode:

In this Mode the instruction specify an address the memory location specify by the address contain the address of an operand

eg → AC = AC + [[X]]

This will Add the operand store at the address specify by the memory location 'X' with the operand of the Accumulator.

(7.) Auto increment or Decrement Mode:

- In this Mode the instruction specify a register which points to a memory Address that contain the operand. After that the address is incremented or Decremented as specify.

8.) Relative Address Mode: In this Mode, the contents of the Address field to the constants store in the Program

Counter. The result of the Addition gives the address of the Operand.

For eg:

Suppose the Address field Contains $50 and 'PC' Contains 20' then the Operand will be at memory location. '$50 + 20' = 870. ∪

9) Index Addressing Mode:

In this Mode the address of the Operand is determined by adding the Content of the Address field and the Content of index Register.

10) Base Register Addressing Mode:

- In this Mode, the address of the Operand is determined by adding the Content of the Address Field and the Content of Base Register.

⇒ DATA TRANSFER AND MANIPULATION.

Data Tranfer:

Computers Provides an extensive set of instruction to give the User flexibility to Carry out various Computational task.

The instruction set of different computer differ from each other mostly in the way the operands are determined from Address & Mode field.

The actual operations Available in the instruction from one Compiler to other.

Data transfer can be classified into 3 categories :-

1) Data transfer instruction
(2) Data Manipulation
(3) Program Controlled instruction.

(1) Data Transfer instruction :

Data Transfer instruction cause of data from one location to another without changing the information content.

(2) Data Manipulation Instruction :

These are those that perform Arithmetic, logic and shift operation.

(3) Program Controlled I/O :

— These provides decisions Making capability and change the path taken by the program when executed in the computer.

Data Transfer operations & Methods

1) LOAD (LD)
2) STORE (ST)
3) MOVE (MOV)
4) Exchange (XCH)
5) INPUT (IN)
6) OUTPUT (OUT)
7) Push (PUSH)
8) Pop (POP)

→ Operations on Data are in above Set table

=> Data Manipulation instruction are also Divided into 3 Categories :-

1) Arithmetic I/s
   - (a) → Increment INC
   - (b) → Decrement DEC
   - (c) → Add (ADD)
   - (d) → Subtract (SUB)
   - (e) → Multiply (MUL)
   - (F) → Divide (DIV)
   - (g) → Add with Carry (ADC
   - (h) → Subtract with Borrow (SUB
   - (i) → Negation (NEG)

(2.) Logical and Bit Manipulation instruction

Clear	CLR
Complement	COM
And	AND
Or	OR
Exclusive OR	XOR
Clear Carry	CLRC
Set Carry	SETC
Complement Carry	COMC
Enable interrupt	EI
Disable interrupt	DI

(3) Shift I/s

1) Logical shift Right	SHR
2) Logical shift LEFt	SHL
(3) Arithmetic shift right	SHR A
(4) Arithmetic shift left	SHL A
(5) Rotate Right	ROR
(6) Rotate left	ROL
(7) Rotate right through Carry	ROR C
(8) Rotate left through Carry	ROL C

(4) Programmed control I/s.

(1)	Branch	BR
2.	Jump	JMP
3.	Skip	SKP
4.	Call	CALL
5.	Return	RET
6.	Compare (By Subtraction)	CMP
7.		
8.	Test (ANDing)	TST

Oct 15, 24

RISC Architecture:

It means Reduced Instruction Set Computing. It is very simple and streamline execution Architecture.

In this Architecture, the I/s set is designed to include a minimum set of simple and frequently used instruction.

Each instruction will perform a specific operation on a small amount of data. This approach enables faster instruction execution. As the processor can complete instructions in a fewer clock cycle.

Features :-

Features of RISC Architecture :—

(1) Simple and Uniform Architecture I/s set with the limited no. of Format.

(2) Single cycle I/s execution.

(3) load store Architecture where memory operation are limited to load and store instructions.

(4) A large number of Register for efficient data [redacted] access.

(5.) Pipelined Execution to achieve higher throughput.

Advantages :

(1). Faster execution due to simple i/s.

2). Efficient use of Register.

3. Well suited for applications like Multimedia processing and Scientific Computing.

4. Simplified hardware design Facility and Faster clock speed.

Disadvantages :

1.) Compler operations may be require more i/s to complete. More i/s may lead to larger program size.

CISC ARCHITECTURE

It Stands For Complex Instruction Set Computing Architecture.

It approach by offering a Rich set of Complex Is that can perform multiple task in a Single Is.

CISC processor aims to reduce the number of instruction require to accomplish a Specific task. Thereby potentially reducing the overall program size and enhancing programer productivity.

Features:

1) Complex instruction capable of Performing multiple operations.
2) A Variety of addressing mode For Flexible memory access.
3) Is may take a More than Single clock cycle to get executed.
4) Less no. of general purpose of Registers as operations get perFormed in memory itself.
5) Complex addressing Mode.

ADVANTAGES:

(1) Reduce Code Size:
CISC Processor use Complex I/s that can perform multiple operation reducing the amount of code needed to perform a task.

2.) More Memory Efficient:
Because CISC Instruction are more Complex they require more. Fewer I/s to perform Complex task which can result in more Efficient Memory.

3.) Widely used:
CISC Processors have been in use For a longer time than RISC Processors. So they have a large use and more available software.

Disadvantages:

1.) Slower Execution:
Because I/s are Complex. CISC Processors take long time to execute and also more time to Decode.

2.) More Complex design:
Because of Complex I/s It is more difficult to design and Manifacture.

(3) Higher Power Consumption:
Use more power consumption because of Complex I/s set.

CPU PERFORMANCES:

Both approaches tries to increase the CPU Performance. RISC: Reduces the cycles per instruction at the cost of the no. of instruction per program.

CISC: It minimise the no. of Is per Program but at the cost of an increase in the no. of cycles per instruction.

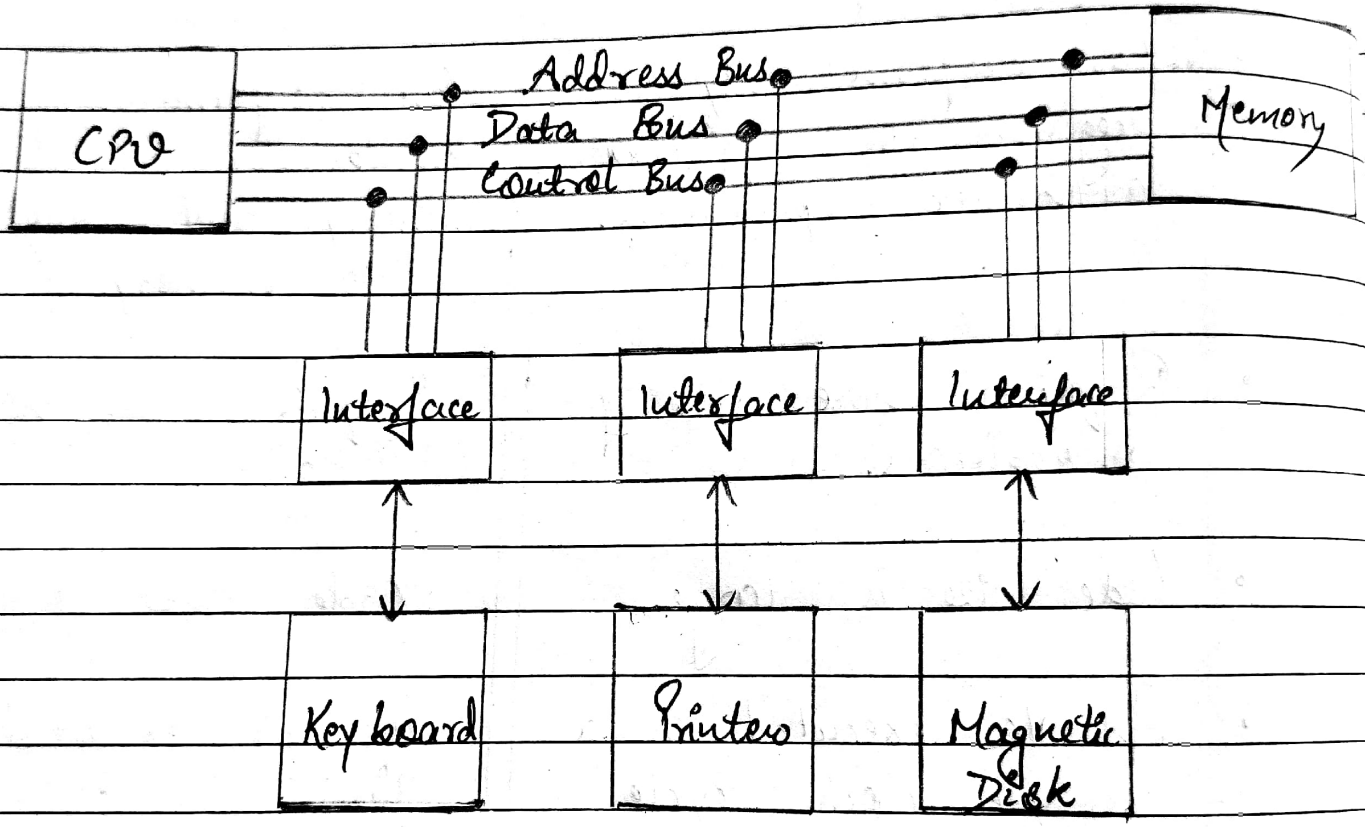
Difference:

RISC	CISC
(1) Focus on Software.	Focus on Hardware.
(2) Transistors are used for more registers.	Transistors are used for storing Complex Is.
(3) Fixed size Is.	Variable size Is.
(4) Can perform only Register to Registers Arithmetic operations	Can perform Register to Register or Register to Memory or also Memory to Memory.
(5) Requires more no. of Register	Less no. of Register is required.
(6) Code size is large.	Code size is small.
(7) An Is Executed in a single clock cycle	More than one clock cycle.
(8) An Is Fit in 1 word.	An Is Fit in larger than the size of 1 word

(9) Simple & limited Addressing Modes	→ Complex & More Addressing Modes.
(10) It consumes the low Power.	CISC consumes high Power.
(11) RISC is highly pipelined	CISC is less pipelined.
(12) Used in Hardwired Control Unit	Used in Microprogrammed Control Unit.

Oct 21; 24

Input - Output Interface



I-O interface

I-O interface is used as a method which helps in transferring information b/w the memory and external hardware devices. A hardware device is that which provide I/P and O/P for the computer. It's also called I-O devices.

for eg. → Keyboard and mouse → I/P devices
Monitor and printer → O/P devices

In micro-programmed computers base system the only purpose of hardware devices is just to provide special communication links for the interfacing them with the CPU and resolve the differences b/w hardware devices and CPU.

The major differences are as follows-

(1) The nature of hardware devices is electromagnetic and electromechanical but the nature of CPU is electronic.
There is a lot of difference in the mode of operations of both CPU and hardware devices.

(2) There is also a synchronisation mechanism bz. the data transfer rate of hardware devices and CPU are different.
Hardware devices are slow than CPU.

(3) In hardware devices data code and formats are differ from the format in the CPU and memory.

(4) The operating mode also different of both devices.
So, there is a special need of

the additional hardware to resolve the difference b/w CPU & hardware devices to supervise and synchronise whole operations.

→ Functions of I-O interface

① It is used to synchronise the operating speed of CPU and I-O devices.
② It selects the I-O devices which is appropriate for the interpretations of the I-O signals.
③ It is capable of providing signals like control signals.
④ It is used for error detection.
⑤ It converts serial data into parallel data and vice-versa.
⑥ It also converts digital data into analog and vice-versa.

→ Modes of data transfer

The primary modes of data transfer are-

① Programmed Input Output (PIO)

In PIO, the CPU actively managed the data transfer b/w itself and I-O device. The CPU checks the status of the I-O device and transfer data b/w memory and device as needed.

Characteristics -

1. CPU involvement in every data transfer.
2. Inefficient for high speed devices and large data transfer.
3. Simple to implement.

(2) Interrupt - driven Input - Output

In this mode, the CPU initiate the I-O operation and then continues the other task. When the I-O operation is complete, the I-O devices generate an interrupt because the CPU suspend its current task and handle the interrupt by servicing the I-O operation.

Characteristics -

1. Asynchronise operation
2. Reduces the CPU involvement during ideal time. (means - CPU is free)
3. Efficient for devices with unpredictable or slow response time.

(3) DMA mode

→ * The main purpose of different modes are-

1. Efficiency
2. Resource utilization
3. Flexibility

Ques What is the purpose of I-O interface?
Ans- To resolve the differences b/w CPU and peripheral devices. Differences occurs on the terms-

1. Speed
2. Signal conversion
3. Format
4. Data words

Advantage of I-O interface →

1. Efficiency
2. Modularity
3. Standardisation

Oct 22, 24

Input - Output Processor

The DMA mode of data transfer reduces the CPU overhead in handling I-O operations. It also allow parallelism in CPU and I/O operations. Such parallelism is necessary to avoid the wastage of CPU time while handling I/O devices whose speeds are

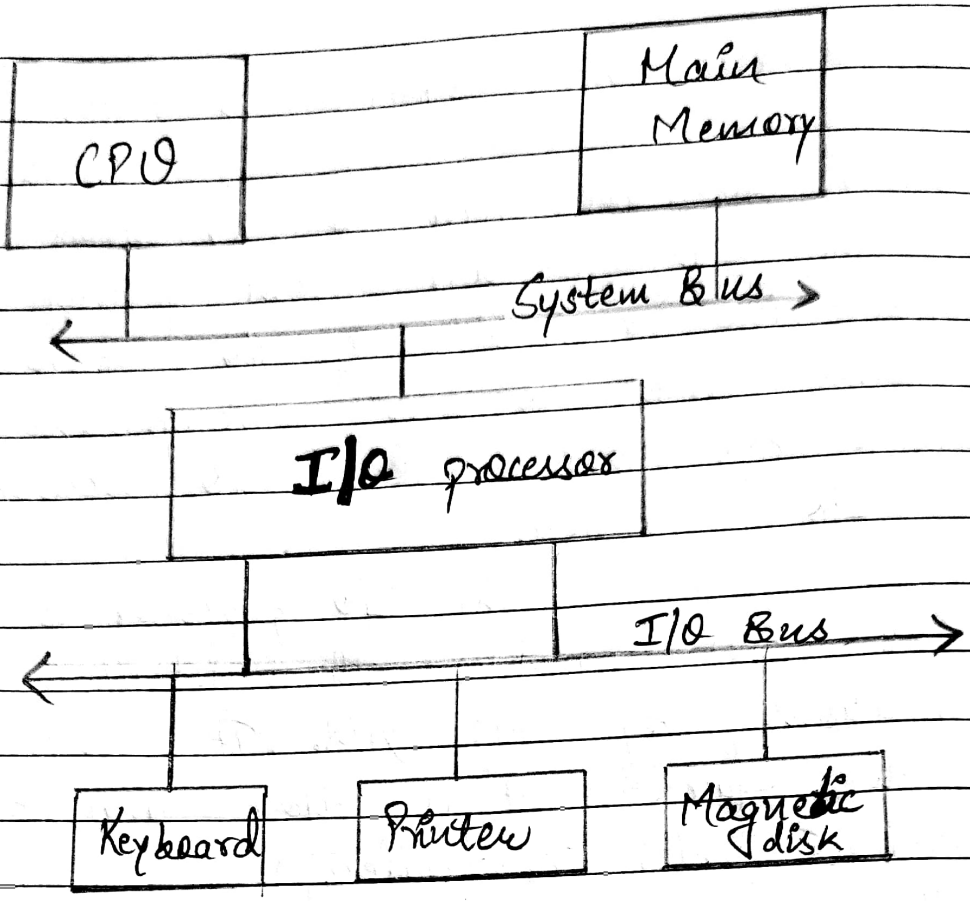
much slower as compared to CPU.

The concept of DMA operations can be extended to relieve the CPU further from getting involve with the execution of I/O operations. This gives rise to the development of special purpose processors called I/O processors or I/O channels.

The I/O processor is just like a CPU that handles the details of I/O operations by the DMA. It's more equipped with facilities than those available in a typical DMA controllers.

The I/O processor can fetch and execute its own instructions that are specifically design to I/O transfers. Also it can perform other processing task like arithmatic, logic, branching and code translation.

Memory communicates with the processor by means of DMA. The I/O processor is specialised processor which loads and stores data in memory along with the execution of I/O instructions. It acts as an interface b/w the system and devices. It involves a sequence of events to execute I/O operations and then stores the result in memory



Features of an I/O processor

1. Specialized hardware
   - I/O ports
   - DMA controller
   - Interrupt controller

2. DMA capability
   - direct communication with DMA and memory without going through CPU

3. Interrupt handling
   - this allows the CPU to focus on executing application programs while the I/O processor handles interrupt.

Date : / /
Page :

④ Protocol handling
- This allows wide range of devices to interface without requiring additional hardware.

⑤ Parallel processing
- This allows the system to handle multiple task simultaneously.

Oct 24 & 29

Serial Communication

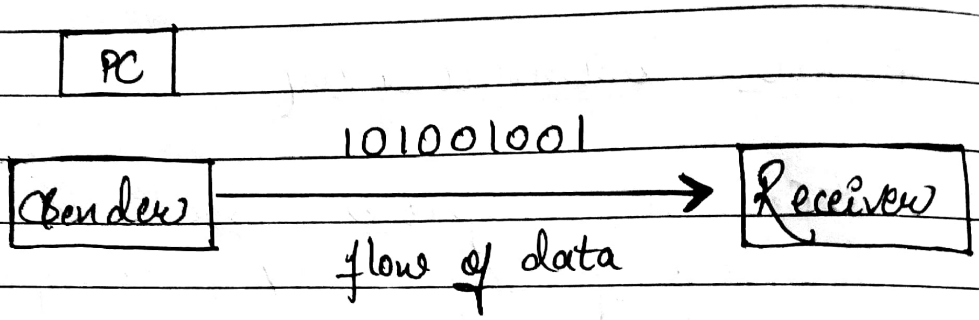
S.C. in computer organisation is the process of sequentially transfering the information or bits on the same channel. Due to this, the cost of wire will be reduced but it slows the transmition speed.

Generally, communication can be described as the process of interchanging information b/w individuals in the form of audio, video, verbal words and written documents.

The serial protocol is run on every device that can be a mobile, PC, and many more. The protocol is reliable and secure that contains a set of rules addressed with the help of a source host and a destination host.

In Serial communication, binary pulses are used to show the data.
Binary contains 2 no. 0 and 1.
0 is used to show the low volts and 1 is used to shows the high volts.

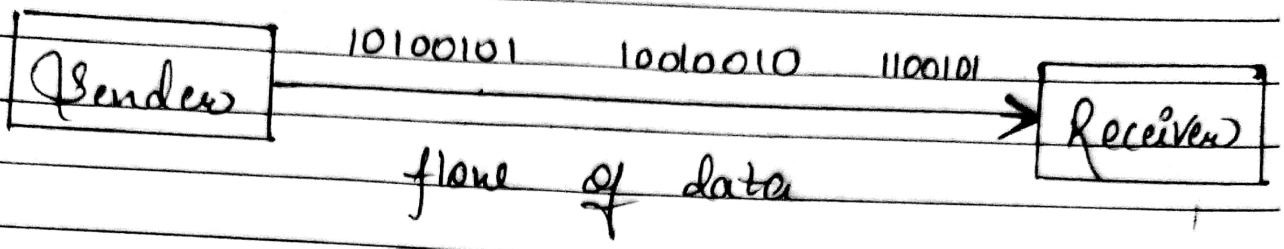
It can be either synchronous or asynchronous.

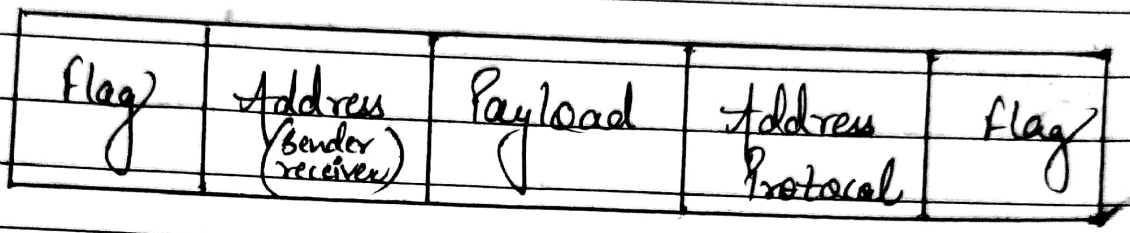


→ Synchronous Communication

In this communication, the frames or data will be constructed with the help of combining the group of bits. That frame will be continuously sent in time with the master clock. It use a synchronised clock frequency to operate the data of sender and receiver. The time taken by the sender and receiver is synced, that's why the frequency of timing error will be less and the data will be more faster. The data accuracy is totally dependent on it.

Synchronous Communication





→ Asynchronous Communication

In this communication, the groups of bits will be treated as an independent unit and these data bits will be send at any point in time. In order to make synchronization b/w sender and receiver, the stock bits and start bits are used b/w the data bytes. These bits are useful to ensure that the data is correctly sent. The time taken by data bits of sender and receiver is not constant and the time b/w transmission will be provided by the gaps.

In this communication, we don't require synchronization b/w the sender and receiver which is the main advantage. This is cost effective and some times the transmission is slow but not compulsory.

UNIT-3

Introduction

Arithematic instructions in digital computers manipulate data to produce result necessary for the solution of computational problems.

These instructions perform arithematic calculations and are responsible for the bulk of activity involved in processing data.

The 4 basic arithematic operations are -

1. Addition
2. Subtraction
3. Multiplication
4. Division

These 4 basic operations formulate other arithematic functions and solve scientific problems by means of numerical analysis method. An arithematic instructions may specify binary or decimal data and in each case the data maybe in fixed point or floating point form.

Fixed point no. may represent integers or fractions. Negative nos. maybe in signed magnitude or signed complement representation. The arithematic processor is very simple if only a binary

fixed point add instructions is included. It would be more complicated if it includes all 4 arithmatic operations for binary and decimal data in fixed point and floating point representation.

OPERATIONS

① Addition and Subtraction

There are 3 ways of representing negative fixed point binary numbers.

1. Signed magnitude
2. Signed 1's complement
3. Signed 2's complement

Mostly we used 2's complement representation when performing arithmatic operations with integers.

But on floating point, most computers use the signed magnitude representation. It's important to realise that the adopted representation for negative numbers refers to the representation of no. in the registers before and after the execution of the arithmatic operation. It doesn't mean that complement arithmatic may not be used in an intermmediate step.

for eg. →

It is convenient to employ complement arithmatic when performing a subtraction

operation with no. in signed magnitude representation. As long as the initial minuend and subtrahend as well as the final difference are in signed magnitude form the fact that complements have been used in an intermediate step doesn't alter the fact that the representation is in signed magnitude.

Addition and Subtraction with signed magnitude data

The representation of numbers in signed magnitude is familiar because we used it in arithematic calculations. The procedure of adding or subtracting 2 signed binary no.'s is simple and straight forward. We designate the magnitude of the 2 numbers by 'A' and 'B'. When the signed no's are added or subtracted, there are 8 different conditions to consider, depending on the sign of the no.'s and the operation performed. These conditions are shown below in the table.

The first column shows the conditions, the other column in the table shows the actual operation to be performed with the magnitude of the no.'s. The last column is needed to prevent a negative zero, in other words when 2 equal no.'s are subtracted the result should be zero, not minus zero (-0).

The algorithm for addition and subtraction are derived from the table.

Addition (subtraction) Algorithm

1. When the sign of A and B are different, Add the two magnitudes and attach the sign of A to the result.
2. When the sign of A and B are different, compare the magnitude and subtract the smaller no. from the larger. Choose the sign of result to be the same as A, if A>B or the complement of sign of A if A<B.
3. The two algorithms are similar except for the sign comparison. If the two magnitudes are equal, subtract B from A and make the sign of the result positive.

Unit-4

Introduction

It means the system achieve data processing task at the same time to increase the computational speed of a computer system.

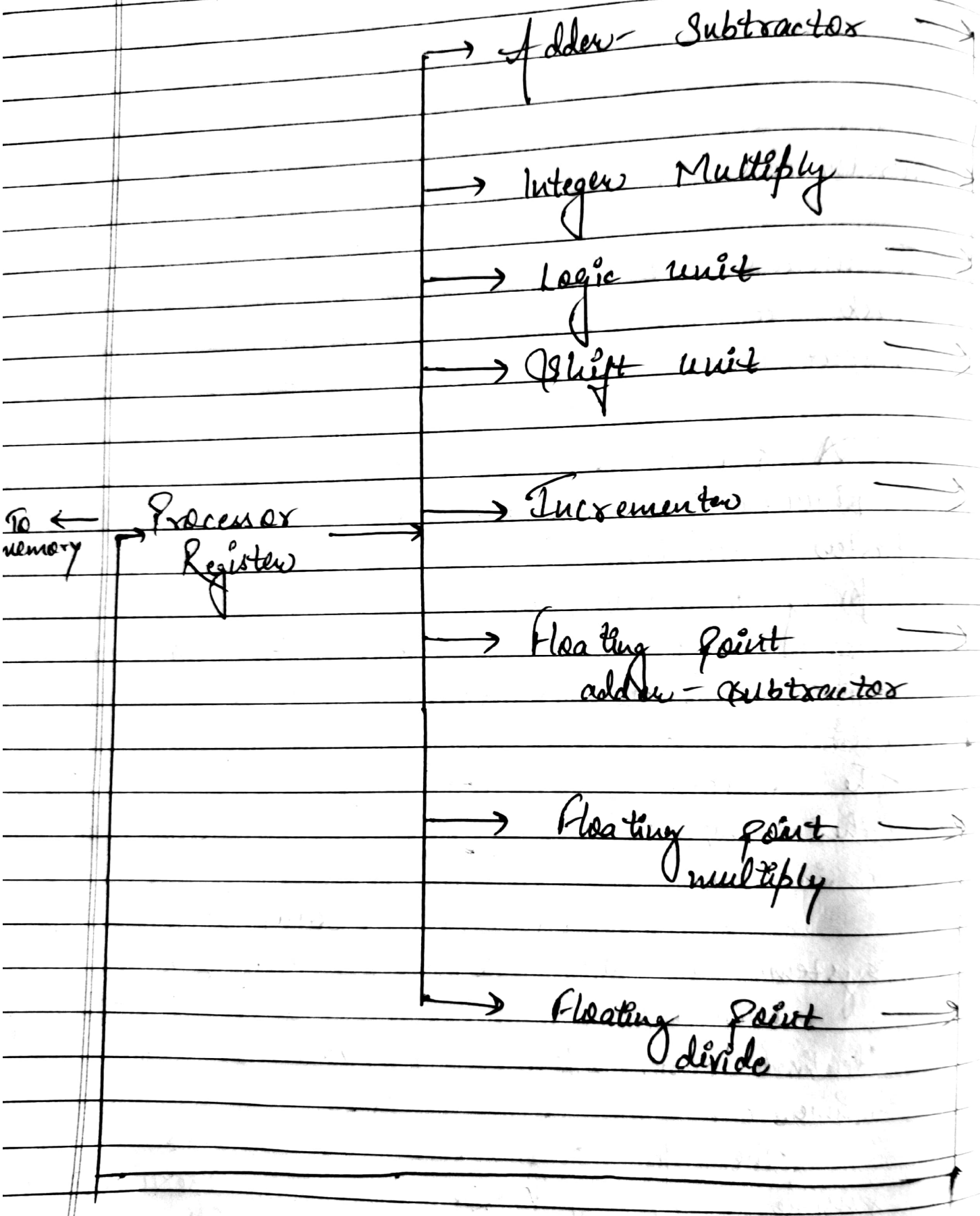
A parallel processing system can carry out simultaneous data processing to achieve faster execution time.

for eg → while an I/S is being processed in the ALU, the next I/S can be read from the memory.

The purpose of II^rd processing is to inhance the computer processing capability and increase its throughput.

The II^rd processing system can be achieve by having a multiplicity of functional unit that perform identical or different operations at the same time.

The following diagram shows one possible way of operating the execution unit into eight functional units operating in parallel.


Adder - Subtractor
Integer Multiply
Logic unit
Shift unit
To memory
Processor Registers
Incrementer
Floating point adder - subtractor
Floating point multiply
Floating point divide

1. The adder and integer multiplier performs the arithmetic operation with integer numbers.
2. The floating operations are seperated into 3 circuits operating in parallel.
3. The logic shift and increment operation can be perform concurrently on different data. All units are independent so one number can be shifted while another number is being incremented.

There are a variety of ways that parallel processing can be classified -

It can be considered from the internal ① Organisation of the processors from ② Interconnection structure b/w the processors or from the ③ flow of information through the system.

One classification is introduced by M. J. Flynn's considers an organisation of computer system by no. of I/S and data items that are manipulating simultaneously.

The normal O/P of the comp. system is to fetch I/S from the memory & execute them in the processor.

* The sequence of I/S read from memory constitutes an I/S stream.

* The operation performs on the data in the processor constitutes a data stream.

Parallel processing may occurs in the I/S stream, in the Data stream or in both.

Flynn's classification divides computers into 4 major groups:

1. Single I/S stream, Single data stream (SISD)
2. Single I/S stream, multiple data stream (SIMD)
3. Multiple I/S streams, Single data stream (MISD)
4. Multiple I/S stream, multiple data stream (MIMD)

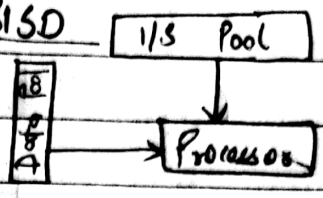
1. SISD

It represents the organisation of a single computer containing a control unit, a processor unit and a memory unit.

I/S are executed sequentially and the system may or may not have internal parallel processing capabilities.

Parallel processing in this case, maybe achieved by means of multiple functional units or by

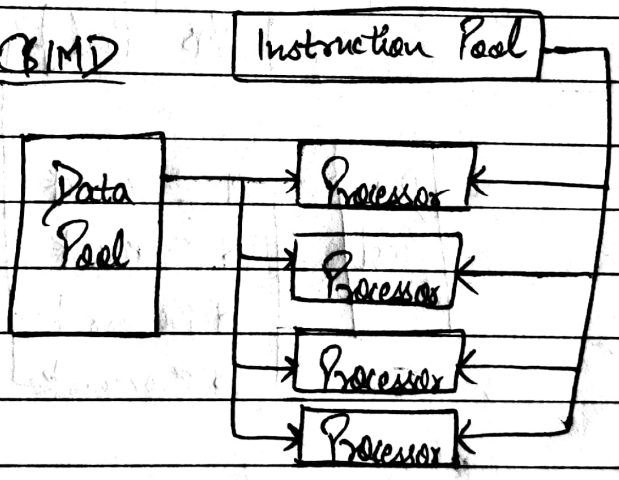
pipeline processing.



2. SIMD

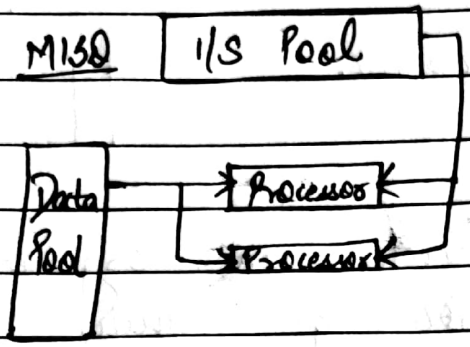
It represents an organization that includes many processing units under the supervision of a common control unit. All processors receive the same I/S from the control unit but operate on different items of data.

The shared memory unit must contain multiple modules so that it can communicate with all the processors simultaneously.



3. MISD

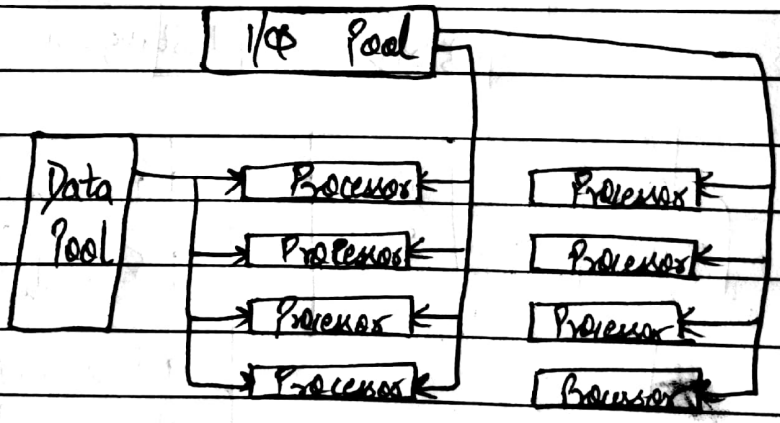
It's a system used multi-processor machine capable of executing different I/S on different processing elements but operating on the same data set. This model is not useful in most of applications.



(more than 1 processor)

4. MIMD

It's a system in which we use multiple I/s on multiple data sets. Each processing element has seperate I/s and data stream. This model is widely used for multiple applications.



Flynn's classification depends on the distinction b/w the performance of the control unit and the data processing unit.

Parallel Processing

It's dividing into 3 categories

- Pipelined Processing
- Vector Processing
- Array Processing

Date : / /
Page :

* Arithematic Pipeline
(used for fast speed)

Pipelined arithematics units are usually found in many high speed computers and used for high speed calculations. Also they are used to implement floating point operations, multiplications of fixed point nos and similar computations.

The inputs to the floating point adder pipeline are to normalized floating point binary numbers.

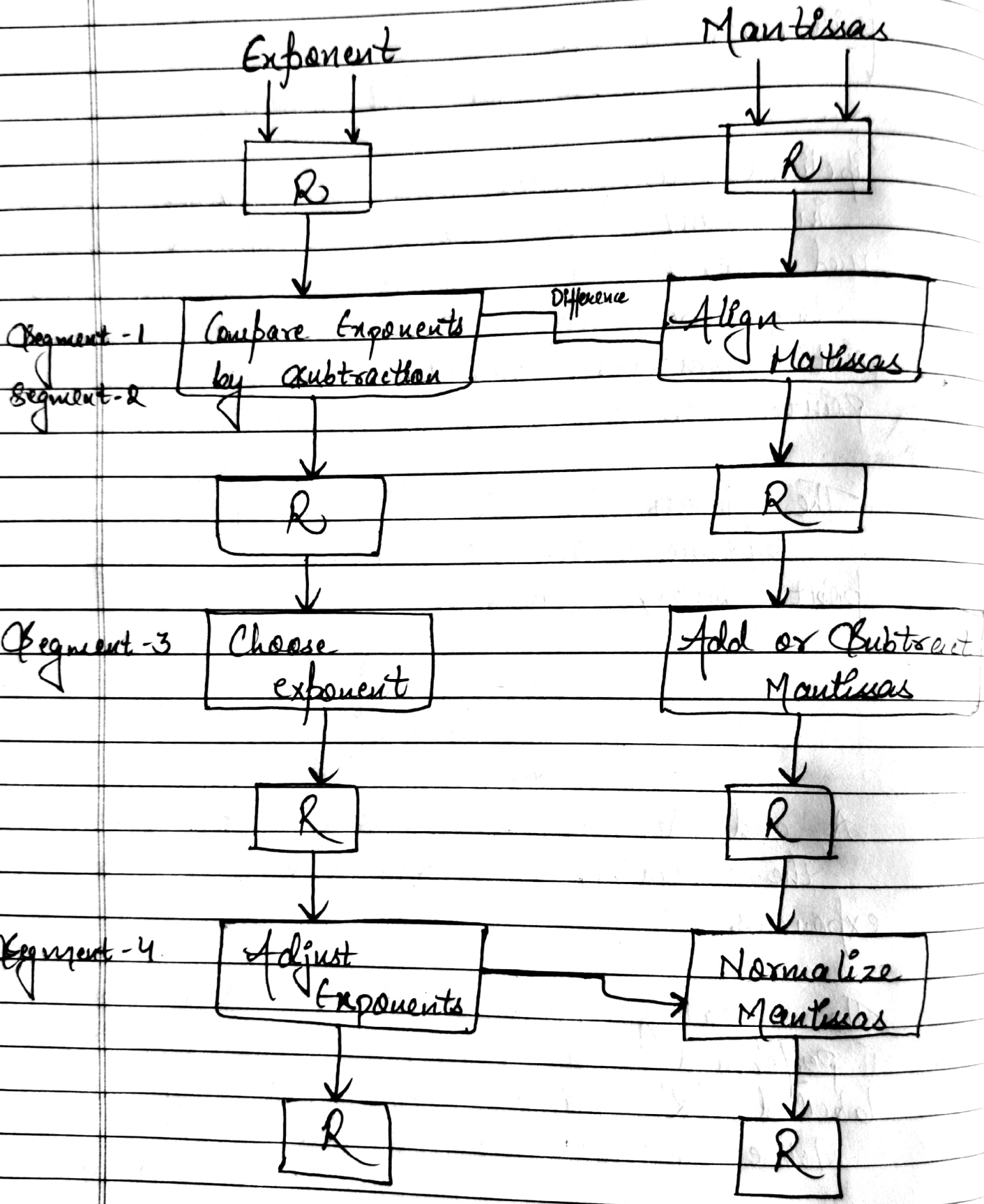
$$x = A \times 2^a$$
$$y = B \times 2^b$$

A and B are 2 fractions that represent the mantissas and a, b are the exponents.

The floating point add" and sub" can be performed in 4 segments. The registers labelled 'R' placed b/w the segments to store intermediate results.

The sub-operations that are performed in the 4-segments are
1. Compare the exponents.
2. Align the mantissas.
3. Add or subtract the mantissas.
4. Normalize the result.

ARITHEMATIC PIPELINE



store in register, $x = 0.9504 \times 10^3$

$y = 0.8500 \times 10^2$

compare, $3 - 2 = 1$ → in register

↳ 1 no. of bits shift/align

align, 0.8500

0.0850×10^3 → in register

add or sub, equal expon – add^n

unequal expon – sub^n

Q) Adding, $0.0850 + 0.9504$

$= 1.0354$

$$\begin{array}{r} 0.0850 \times 10^3 \\ +\ 0.9504 \times 10^3 \\ \hline 1.0354 \times 10^3 \end{array}$$ → in register

normalise, .----

↳ ≠0 normalised

↳ =0 not normalised

Q)

align, 1.0354×10^3

$\Rightarrow 0.1035 \times 10^4$ → in register

normalised, 0.1035×10^4